ANDREW SEELY

building a virtual DNS appliance using Solaris 10, BIND, and VMware

Andy works for Science Applications International Corporation as the Senior Command and Control UNIX Engineer at USCENTCOM-J6, and after hours he teaches computer studies courses for the University of Maryland University College. His wife Heather is his init process and his son Marek's first word was "#!".

seelya@saic.com

I WAS FACED WITH THE TASK OF CREATing a new DNS server solution that would be simple to build and maintain, would be easy to deploy to remote locations, and would bring the site into compliance with DoD security requirements. The goal was to improve performance and maintainability while diversifying the DNS architecture and to accomplish it for free. The tools on the table were BIND, Solaris 10, and the site's existing VMware ESX server installation. In this article I explain the background and requirements and detail the installation procedure and scripts developed to deliver a virtual appliance solution.

A Unique Customer with Unique Challenges

I work for Science Applications International Corporation (SAIC) on contract to support the Global Command and Control System (GCCS) at the HQ US Central Command (CENTCOM) in Tampa, Florida. The GCCS shop has the majority of UNIX servers in the organization, so I am also the lead UNIX system administrator. Our contract overall supports the entire spectrum of systems support for the Command, from configuration management to systems engineering, from service desk to cable installation. You'll find members of our team in Tampa writing custom military applications software, in Kabul configuring routers, and everything in between, all in support of the CENTCOM mission.

CENTCOM, like many other government organizations [1], is rapidly expanding its use of virtualization, with a particular focus on server consolidation due to limited datacenter space. If it's a new system coming into the building, the first thing that gets asked is if it can be virtualized. CENTCOM also employs network appliance technologies when appropriate, especially when the technology may be intended for a remote site with disadvantaged communications and limited on-site technical skills. New technical services for the Command will be considered for virtualization, for viable appliance solution, and for commercial off-the-shelf (COTS) purchase. In some cases, new requirements will be candidates for in-house development. In rare circumstances, a solution touches all four.

An appliance-oriented approach to virtualization can bring together the benefits of network appliances and virtualized servers while reducing risk

and cost. This is the approach we took when faced with a customer challenge to improve Domain Name Service (DNS) architecture. By leveraging the latest Solaris 10 x86, Berkeley Internet Name Domain (BIND) software, and the Command's existing VMware ESX environment, a DNS appliance-equivalent was developed, tested, and brought into service for external authoritative and internal recursive DNS requirements.

Motivators for Change

DNS is the kind of service that can start out easy and, without significant effort, scale over time to support an expanding organization. But without a roadmap, DNS can devolve into a patchwork that no one understands, with inherent fragility that can baffle even the most seasoned sysadmin. This was the situation at our customer site. DNS had been grown rather than planned and, due to the natural turnover of military leadership over the course of years, there was no single guiding authority or principle to direct that growth; in military terms, there was no Concept of Operations (CONOPS). DNS worked fine for years, until suddenly it didn't work at all for a few days. The entire site was effectively down while the operations team essentially had to spelunk the DNS configuration to find the failure point and repair it. While workarounds were immediately implemented to prevent mission compromise during the outage, all involved understood that a military network is a vital tool and that this type of fragility is unacceptable.

One of the findings that surfaced out of the after-action review was that the site's DNS architecture was not aligned with industry best practices and the Defense Information Systems Agency (DISA) Security Technical Implementation Guide (STIG) [2] requirements for split-DNS and heterogeneous implementations. During a redesign discussion it was noted that both problems could be fixed at once by deploying a new and different set of DNS servers for the relatively simple external authoritative and internal recursive roles, leaving the more complex internal authoritative zones to be served by the site's existing Microsoft implementation.

Classic Build-Versus-Buy Decision

There are proven DNS solutions in the appliance market, with a focus on security, ease of use, and total cost of ownership [3]. These appliances tend to use a hardened Linux kernel and a purpose-built Linux distribution to reduce the total number of system "knobs that may be turned." The Infoblox [4] appliance was strongly promoted at the redesign meeting as the best way to simplify management and integration. Others at the table were anti-appliance, reminding leadership that the failure case for an appliance means a field engineer service call and expensive support contracts compared to in-house expertise that can respond immediately. This caused everyone to discuss what it would mean to build something in-house. Everyone agreed that BIND on Solaris would be an ideal solution from a performance standpoint but was still not as attractive as the appliance from a security and simplicity point of view. I raised a hand and volunteered to build BIND on Solaris, virtualize it in VMware, meet all the factors that made the appliance appear attractive, and do it at no cost.

The DNS "virtualized appliance" project resulted in an appliance-like DNS capability that allows the Command to comply with DNS security requirements, simplifies overall configuration, and significantly improves configuration management control of the external authoritative DNS function. Total cost of ownership may ultimately be higher due to the relatively few Solaris

experts on the team, but clear documentation of design and the very terse installation leave very few moving parts to debug and should prevent any perception of added expense. Using the virtualization technique, we were able to save a little bit of space and power in an already cramped datacenter, and by focusing on security issues as a design parameter I was able to make a hardened solution that easily conformed to Department of Defense (DoD) requirements for UNIX systems [5].

Initial Design Parameters and Assumptions

The Command leadership's biggest complaint (after, of course, DNS having suffered an outage) was that configuration management (CM) of the DNS was almost nonexistent. Multiple people in different shops had access to zone files, there was little accountability for changes, and there was no revision control. For serious CM, the DNS appliance technical solution needed to be considered from the start.

Command Information Assurance (IA) immediately had concerns about STIG-compliance and overall accreditation of this type of system. Without IA approval, no solution, no matter how good, would see a network light. The project needed to be STIG-compliant as it grew, preferably with IA involvement at each step of the development.

The solution had to be small, agile, and portable, easily deployable to other networks, and easy to upgrade and maintain. We developed this into a DNS "multiple master" concept, with a plan for potentially multiple DNS master authoritative servers without any slaves, increasing the simplicity of design and deployment while obviously trading off the flexibility inherent in a master-slaves architecture. We determined that there would be no requirement for any BIND server cross-talk from other areas of the DNS architecture, which effectively eliminated any requirement for an authorization list or key management and made the product highly and rapidly deployable. There were very few tunable knobs in the design parameters.

The Technical Approach

To limit the attack surface of the system we agreed that the only networking ports that would be available would be UDP/53 and TCP/53 in and out for processing DNS requests, and UDP/123 out for Network Time Protocol (NTP). When a syslog host was implemented, then UDP/514 would need to be opened between the DNS host and the syslog host. Otherwise, all network ports would remain closed, even TCP/22. All command-line access was limited to console-only and would be controlled and audited tightly by access from the VMware console.

To keep the footprint as small as possible, I installed the system with Solaris 10's minimum possible disk allocation of 1.2GB. This could be shrunk down after the operating system was installed, but there was no requirement to have a smaller installation.

My initial development environment was VMware Workstation 6.5.3. I added a new VM with parameters for Workstation 5 and ESX server compatibility enabled. The initial concept was to build the vmdk image file in VMware Workstation and then import it into ESX. This worked well for initial proof-of-concept testing, but for production I built from scratch in the native environment.

I built a Solaris 10 U8 system with ZFS, no naming service defined and no remote services enabled. I selected the "Reduced Networking Core System

Support" group and customized it to remove unnecessary tools and to add in essential options:

- Remove audio drivers and applications
- Select BIND nameserver manifest
- Select Basic Audit Reporting Tool (BART)
- Select Basic IP Commands (root)
- Select Basic IP Commands (usr)
- Remove Kerberos version 5 support
- Remove Network Information System (NIS) support
- Select Network time protocol
- Select Programming Tools
- Expand Remote Network Services and Commands and select Remote Network Client Commands
- Select secure shell
- Remove non-mandatory Universal serial bus software
- Remove libexpat
- Remove xvm paravirtualized drivers

This configuration creates dependency warnings for USB, GSSAPI v2, and Kerberos. These warnings may be safely ignored. This build process results in a very terse Solaris 10 installation. But there's more work to be done to make it ready for action.

In a separate VM I installed a full OEM Solaris 10 with all defaults as a development environment. In that environment I installed the Solaris Freeware GNU C compiler [6] and all related dependencies. In the development environment I built several scripts and archives to make deployment of the DNS appliance rapid, simple, and controlled. The pre-configured system files I set up were:

- profile for root to set umask, mesg, stty, and TMOUT values for STIG compliance and usability
- syslog.conf to set standard site logging specifications
- Customized DNS server manifest to set chroot for BIND [7]
- resolv.conf to set the site domain and the local host as the nameserver
- ntp.conf with the site's NTP server IP, and defined locations for driftfile and statsdir
- motd to meet DoD and STIG requirements
- db.0.0.127.in-addr.arpa.txt with a typical loopback PTR record
- db.roothints.txt with DoD root servers
- named.conf: The initial default configuration is for a recursive server only; authoritative installations will require the named.conf and zone files to be updated. Set logging options for syslog to support a remote log host.
- bart.rules, with directories to be monitored by the Solaris 10 Basic Audit Reporting Tool (BART) [8]: /var/named, /etc, /usr, /opt, /bin, /boot, /sbin, /lib
- lock: A script (see Listing 1, below) to disable useful tools before a host is placed into production. Note that whenever the machine is locked down, BART is executed, and a sysadmin is expected to follow up on any BART-reported changes before the system goes live.

```
#!/bin/sh
svcadm disable ftp
svcadm disable ssh
chmod 400 /usr/bin/truss
chmod 400 /usr/sbin/ping
chmod 400 /usr/sbin/traceroute
chmod 400 /usr/bin/ftp
chmod 400 /usr/bin/telnet
chmod 400 /usr/bin/ssh
```

```
chmod 400 /usr/sbin/snoop
chown –R root:root /root
chmod –R o-rwx,g-rwx /root
if [ -d bart ]
then
echo Creating BART baseline
cd bart
latest=`ls -tr1`
rightnow=bart.`hostname`.`date %Y%j%H%M`
bart create -r bart.rules > $rightnow
echo Comparing BART baseline
bart compare $latest $rightnow
[ $? -ne 0 ] && echo BART discrepancies found || echo BART integrity OK
cd ..
fi
```

LISTING 1: THE LOCK SCRIPT DISABLES SERVICES, CHANGES PERMISSIONS TO ROOT ONLY FOR SOME COMMANDS, AND RUNS BART.

- unlock: A script that enables the services and utilities that are disabled by the lock script.
- recursive.tar: A tar of the /var/named directory, used to quickly recreate the chroot file environment on the production system.
- configure.sh: The script (Listing 2) to execute on the production build that completes the configuration and makes the host ready for operations.

```
#!/bin/sh
groupadd named
useradd -m -d /var/named -c "BIND User" -s /bin/false -g named named
tar -xf bind.binaries.tar
tar -xf recursive.tar
cp syslog.conf /etc/syslog.conf
cp .profile /root/.profile
cp motd /etc/motd
cp ntp.conf /etc/inet/ntp.conf
cp server-chroot.xml /var/svc/manifest/network/dns/
cp resolv.conf /etc/resolv.conf
mkdir -p /var/named/var/named
mkdir /root/bart
mkdir /var/named/var/log
mkdir /var/named/var/run
cp /etc/rndc.key /var/named/etc
chown -R named:named /var/named
mkdir /var/named/dev
mknod /var/named/dev/poll c 135 0
chmod 666 /var/named/dev/poll
chmod 640 /var/named/etc/named.conf /var/named/var/named/*
svccfg validate /var/svc/manifest/network/dns/server-chroot.xml
svccfg delete dns/server
svccfg import server-chroot.xml
svcadm enable dns/server
svcadm enable ntp
svcadm disable network/inetd
svcadm disable cron
svcadm disable name-service-cache
svcadm disable iscsi/initiator
svcadm refresh system-log
```

LISTING 2: THE CONFIGURE.SH SCRIPT SETS UP BIND AND DISABLES UNNECESSARY NETWORK SERVICES.

In order to keep the production system as clean as possible and maintain the ability to independently upgrade the BIND version without waiting for Solaris patches, I did an offline compile and transfer. In the development environment, I downloaded the latest BIND source distribution from ISC [9] and compiled with prefix of /usr and sysconfig of /etc. After compiling and testing, I made a tar roll-up into bind.binaries.tar of all the newly created binaries from /usr/sbin, /usr/lib, and /usr/bin.

Finally, I made a single config.tar.Z containing all the files created in the development environment. This is the "secret sauce" that is required to complete the transition from generic "terse Solaris 10" to hardened DNS appliance. To complete the configuration, enable ssh, set up root's $HOME, run the configuration script, and run the lockdown script:

Enable ssh for root. This is required to copy the configuration file in; ssh will be disabled in the lock script before the host is deployed in a live environment.

```
/etc/ssh/sshd_config, change "PermitRootLogin no" to "PermitRootLogin yes"
svcadm refresh ssh
scp config.tar.Z to /root/config
```

Set up root's home directory:

```
mkdir –p /root/config
Edit /etc/passwd, change ":/:" to ":/root:"
```

Configure the host and lock it down:

```
cd /root/config
uncompress config.tar.Z
tar –xf config.tar
rm config.tar
./configure.sh
./lock
```

The result is what I consider to be a "versioned release." Only copies will be given live IP addresses and placed into production, while a read-only archive of this and future versions will be made for reference. To turn this into an authoritative DNS server the named.conf and zone files must be updated to reflect the authoritative zones and to remove the root hints reference to prevent recursive queries.

A Close Look at the Running System

The resulting DNS appliance presents a very low-drag surface. Required VM server farm resources are limited to 2GB of disk device storage and 1024MB of system memory per installation. The only actively listening port is for BIND and the system is protected from unauthorized network connections in or out by an external firewall monitored by the security team. System boot time in the VM is less than a minute. DNS response time is rapid and scales well; using first perfquery and then a scripted local test harness we successfully tested the installation under loads exceeding that experienced by the previous DNS. The only processes running are essential kernel functions, ntpd, and named, as shown in the following process table:

```
# ps -ef
     UID   PID  PPID   C    STIME TTY      TIME CMD
    root     0     0   0 18:31:17 ?    119:31 sched
    root     1     0   0 18:31:18 ?     0:00 /sbin/init
    root     2     0   0 18:31:18 ?     0:00 pageout
    root     3     0   0 18:31:18 ?     0:00 fsflush
    root   240     1   0 18:31:29 ?     0:00 /usr/sbin/syslogd
    root     7     1   0 18:31:19 ?     0:02 /lib/svc/bin/svc.startd
    root     9     1   0 18:31:19 ?     0:05 /lib/svc/bin/svc.configd
  daemon   130     1   0 18:31:26 ?     0:01 /usr/lib/crypto/kcfd
    root   201     7   0 18:31:28 ?     0:00 /usr/lib/saf/sac -t 300
    root   222     1   0 18:31:29 ?     0:00 /usr/lib/utmpd
    root   336   232   0 18:44:34 console 0:00 ps -ef
    root   232     7   0 18:31:29 console 0:00 -sh
    root   110     1   0 18:31:25 ?     0:00
/usr/lib/sysevent/syseventd
    root   263     1   0 18:31:32 ?     0:00 /usr/lib/inet/xntpd
    root   205   201   0 18:31:28 ?     0:00 /usr/lib/saf/ttymon
   named   267     1   0 18:31:33 ?     0:01 /usr/sbin/named -t
/var/named
    root   248     1   0 18:31:30 ?     0:03 /usr/lib/fm/fmd/fmd
#
```

The CM Plan

One of the biggest concerns leadership had with the original DNS way of doing business was the lack of configuration management or attribution for system changes. By using the VMware images, a "golden image" concept, extremely limited direct access, and a strict workflow, the DNS zones will be extremely stable. The key to this approach is that all configuration changes will be made offline to a candidate release; only after the candidate release is tested and validated will it be cloned for production and then archived for reference. Candidate change lists will be approved by the DNS, UNIX, and Security teams before implementation. The planned battle rhythm is:

- Monthly: New numbered release incorporating updated DNS configurations, security patches, antivirus updates as required by DISA standards [5, 10], and other approved configuration changes. Root password is changed monthly.
- Annually: New major-number release for major operating system updates, as released.
- As needed: Independent security scans as new scanning tools are updated.
- As needed: Emergency updates for security patches or DNS configurations.

What About User Accounts?

In the DoD space, passwords have extreme complexity, length, and change frequency requirements. Given that each deployment of the DNS appliance is expected to stand alone and that direct access will be limited by the VMware console, a conscious decision was made to completely remove the requirement for user accounts and thus the requirement for accounts management. As delivered, this virtualized DNS appliance is limited to the root login with user audit requirements satisfied at the VMware console.

Part of the roadmap for the DNS appliance project is to completely lock the root account, effectively removing *any* command-line access from an operational system. This remains a controversial topic with most of the site's system administrators, but consider the possible needs for logging in to the operational system:

- Inspect system logs: No need because logging is sent to a syslog host.
- Clear file systems: The limited applications running on this system do not generate files and logs are sent to the loghost.
- Performance monitoring: Utilization of CPU, memory, and I/O all are accomplished from the VMware hypervisor monitor.
- Firewall log analysis: Port-level protection has been outsourced to the external firewall, where monitoring is already being performed.
- Performance tuning or zone file editing: All system changes should be done in a candidate release image that gets tested, cloned, and swapped out with the running system; there should never be an edit of the configuration of a running system.
- General troubleshooting: In the event that a system needs troubleshooting, a new, unlocked clone will be put on the live network to gather data and then taken back offline for analysis.
- Log in to change passwords every 90 days: No need to do this if there are no enabled accounts on the system!

Results of the Virtualized DNS Appliance Project

There is no maintenance or licensing sustainment fee, no additional charges for increasing the installation base, and the only cost of sustaining ownership is the maintenance of a Solaris skill set within the Command. Solaris 10 is used with a cost-free license, BIND is used with a cost-free license, and the existing VMware ESX covers the expanded use. Total additional cost to the customer: $0.

The DNS appliance project produced a hardened, reliable, scalable, DNS solution that meets all design parameters and is compliant with DoD and Command information assurance requirements. The appliance concept has been deployed operationally for eight DNS server installations, without any loss or degradation of service, and more deployments are expected as the requirement grows and the product matures.

REFERENCES

[1] "Virtualization and Consolidation in the Federal Government," *Government Computer News*: http://gcn.com/microsites/virtualization-consolidation/efficiency-gains.aspx.

[2] DNS STIG: http://iase.disa.mil/stigs/checklist/unclassified_dns_checklist_v4r1-8_20100226.doc.

[3] "Do It Yourself DNS," *Network Computing*: http://www.networkcomputing.com/1406/1406f3.html.

[4] Infoblox: http://www.infoblox.com/.

[5] UNIX STIG: http://iase.disa.mil/stigs/checklist/unclassified_unix_checklist_v5r1-23_20100226.zip.

[6] Solaris freeware compiler: http://www.sunfreeware.com/indexintel10.html.

[7] BIND chroot manifest example: http://blogs.sun.com/anay/resource/server-chroot.xml.

[8] Basic Audit Reporting Tool (BART): http://docs.sun.com/app/docs/doc/816-4557/bart-1?a=view.

[9] BIND download site: http://ftp.isc.org/isc/bind9/.

[10] Jim Laurent's commentary on the DISA requirement for UNIX antivirus: http://blogs.sun.com/jimlaurent/entry/update_anti_virus_software_for.