An Interesting DNS Solution

By David Bank/Copyright (c) 2019-2023 David Bank

Here's an interesting DNS problem I ran into, and my solution for it.

A Name by any other Rose is still a Name…

The environment was SLES for SAP v15 on PowerPC (specifically, IBM P980s), with some also running under VMware. For various reasons outside of the scope of this paper, the environment had a dedicated "build network"; a separate VLAN and IP address space in which new LPARs and VMs were initially loaded. Also, and again for various reasons, I wasn't deploying using images. The SUSE Manager (SUMA) host provided network-based installations, using the build network; and I was using the "Traditional", instead of SALT-based, management.

Once the OS was laid down *via* **AutoYaST**, then in the "Traditional" environment the *bootstrap* script would "register" the host during the **Firstboot** process (that is, the first time the host is booted following installation). In order to do that, it must talk to the SUMA, and at that point in time it had to do so across the "build network".

As the "build network" was truly isolated, the "normal" DNS services were not available within. That posed some issues.

Certainly, it was possible to refer to the SUMA by the IP address it had in the "build network". The trouble with that was when the host registered, its **mgr-daemon** configuration would contain that IP - the tools will never attempt to resolve the name, and once the host no longer had any connectivity to that isolated "build network", the tools would lose all contact with SUMA.

If I had the build process refer to the SUMA by its hostname, then that created two problems: first, name resolution services had to be offered in the "build network"; second, if the DNS request was merely relayed to the organization's usual nameservers, then the reply would not be the "build network" IP address for the SUMA, but rather the "public" or "management" IP – and, again, the host-under-construction would be unable to resolve the name, and the tools would lose contact with the SUMA while still finalizing the build.

In another environment, I (potentially) would have been able to address that situation *via* routing. But that wasn't optimal for me at the time, and so I ended up engineering a solution where I could let the **FirstBoot** process register the host to SUMA using the hostname (which, of course, meant I did need to provide DNS services within the "build network"). Because the issue was very narrow in scope (basically, affecting just SUMA), I decided that DNS Views was a bit of overkill.

For that environment, I designed an approach that I find easier to implement and maintain. On the SUMA, I ran a DNS server (BIND v9) that had no function other than to resolve its hostname to the IP address it had in the "build network". I forwarded all other resolution requests to my organization's main nameservers for normal resolution.

Here are my configurations:

```
# /etc/sycsconfig/named
################################################################
# This file defines how SLES generally manages the BIND service
#
# IMPORTANT! It does NOT define the configuration of named !!
#                                The daemon itself is configured via /etc/named.conf
################################################################
# Change Log (Reverse Chronology)
# Who    When_____      What_____________________________________
# dxb    2019-04-18     Initial creation
################################################################
# Run BIND server in a chroot jail (/var/lib/named)?
# Valid Values:    no = do NOT run named in a Jail
#                  yes (DEFAULT) = Jail named
# If set to "yes", then each time named is (re)started, /etc/named.conf is
#          copied to /var/lib/named
# Also, other configuration files (/etc/named.conf.include, /etc/rndc.key, and
#          all files listed in NAMED_CONF_INCLUDE_FILES) are copied to
#          the chroot jail
# The PID file will be /var/lib/named/var/run/named/named.pid
NAMED_RUN_CHROOTED="yes"

# Additional arguments for the named server command-line
# Specified as a string of valid options (default is an empty string)
# Example: "-n 2" = use two CPUs (helpful if named is unable to determine
#                                the number of available CPUs)
# NOTE: If NAMED_RUN_CHROOTED="yes", then
#                  "-t /var/lib/named/var"is added to the string
NAMED_ARGS=""

# Additional BIND configuration files
# Specified as a space-separated string of either files or directories (omit
#          trailing /); if a relative path is provided, it is relative to /etc/named.d/
# Default is an empty string
# Example: "/etc/bind-dhcp.key ldap.dump"
# NOTE: /etc/named.conf, any files mentioned as includes in that file, and
#          /etc/rndc.key are all always copied regardless of this setting
NAMED_CONF_INCLUDE_FILES=""

# Define other programs that are executed every time named is (re)started
# Specified as a space-separated string of files; relative paths to begin
#          with /usr/share/bind/
# Default: "createNamedConfInclude"
NAMED_INITIALIZE_SCRIPTS="createNamedConfInclude"

# End of /etc/sycsconfig/named
################################################################
```



```
//////////////////////////////////////////////////////////////
# /etc/named.conf
# NOTE: This file is COPIED to the chroot jail when BIND starts, so be
#                careful that you edit the correct copy
//////////////////////////////////////////////////////////////
# Change Log (Reverse Chronology)
# Who   When_____     What_____________________________________
# dxb   2019-04-18    Initial creation
//////////////////////////////////////////////////////////////
# Mostly, defaults are used; exceptions are "listen-on", "listen-on-v6" and the
#         addition of a Zone for "domain.tld" - the Zone file should exist as
#         /var/lib/named/<domain>.<tld>.zone
//////////////////////////////////////////////////////////////
options {
        # Define BIND's working directory
        directory "/var/lib/named";

        # Enable DNSSEC & validation?
        # Valid Values:   no = Disable
        #                 yes = Enable
        # For dnssec-validation only
        #                 auto = (DEFAULT) let named decide
        dnssec-enable no;
        dnssec-validation auto;

        # Set the keys directory
        managed-keys-directory "/var/lib/named/dyn/";

        # Write dump and statistics file to the log subdirectory
        # Pathnames are relative to the chroot jail
        dump-file "/var/log/named_dump.db";
        statistics-file "/var/log/named.stats";

        # The forwarders record contains a list of servers to which queries
        #         should be forwarded; up to three IPs may be listed
        forwarders { 10.0.1.200; 10.0.2.200; };

        # Should BIND forward instead of attempting to resolve locally?
        # Valid Values:   first = Forward, then attempt to resolve locally if needed
        #                 no = Never forward
        #                 yes (DEFAULT) = Attempt local resolution, then forward
        # forward first;    # I used the default
```



```
        # Define a list of local IPv4 network interfaces to listen on;
        #       optionally the port can be specified
        # Valid Values: 'any', 'none' or a list of addresses
        # Default: 'any' (default port is 53)
        # In my environment, this was the IP of the SUMA interface in the
        #       “build” network
        listen-on port 53 { 192.168.1.10; };

        # Define a list of local IPv6 network interfaces to listen on
        # Valid Values: 'any', 'none' or a list of addresses
        # Default: 'any'
        # In my environment, we were not using IPv6
        listen-on-v6 { none; };

        # Define a list of IPv4 network addresses allowed to submit
        #       queries, and recursive queries
        # Valid Values: 'any', 'none' or a list of addresses
        # Default: 'any'
        allow-query { any; };
        allow-recursion { any; };

        # The allow-query record contains a list of networks or IP
        #       addresses from which queries are allowed or denied
        # allow-recursion does the same for recursive queries
        # Default is to allow queries/recursion from anywhere

        # Should other name servers get notified if a Zone changes
        # Valid Values:
        #       yes (DEFAULT)
        #       no
        # In my environment, this was not needed
        notify no;

        disable-empty-zone "1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.IP6.ARPA";
};

# Logging is best-handled on a per-site basis and so does not appear here

# The following FOUR zone definitions should not need any modification
zone "." in {
        type hint;
        file "root.hint";
};
```



```
# Zone for localhost
zone "localhost" in {
        type master;
        file "localhost.zone";
};

# Reverse-lookup zone for IPv4
zone "0.0.127.in-addr.arpa" in {
        type master;
        file "127.0.0.zone";
};

# Reverse-lookup for IPv6
zone "0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.arpa" IN {
        type master;
        file "127.0.0.zone";
};

# Finally, the Zone for the SUMA – the whole point of this paper
# The Zone file should exist as
#       /var/lib/named/<FQDN of your SUMA>.<your Domain>.<tld>
zone "<FQDN of your SUMA>.<your Domain>.<tld>" in {
        type master;
        file "<FQDN of your SUMA>.<your Domain>.<tld>.zone";
};

# Include the meta include file generated by createNamedConfInclude.
#       This includes all files as configured in
#               NAMED_CONF_INCLUDE_FILES
#       from /etc/sysconfig/named
include "/etc/named.conf.include";
# End of /etc/named.conf
################################################################
```

And now, at long last, the Zone file:

```
; <FQDN of your SUMA>.<your Domain>.<tld>.zone
; Zone file to allow hosts in the Build network to resolve the SuSE Manager
;       server host name

; Set a TTL for this Zone to 1 day
$TTL 1D

$ORIGIN <FQDN of your SUMA>.<your Domain>.<tld>.
@ IN SOA @ ns (
```



```
                2019041801; serial #-increment for changes
                2H ; refresh
                4M ; retry
                1H ; expiry
                1H ) ; minimum

; Apex record for the Zone
@ IN A 192.168.1.10

; Name server info for zone
@ IN NS ns
ns IN A 192.168.1.10
```

Again, the purpose of this configuration was to create an BIND instance that would forward almost all requests, considering itself " authoritative" for a single host that I wanted to resolve to a specific IP for queries in a specific, otherwise isolated network. If I'd controlled the entire DNS infrastructure for the environment and was using BIND for it too, then I could have done this with Views, but that wasn't the case.

If you find any of these ideas applicable to your environment, you're welcome to use them.