

# NeXT System Administration Release Notes: NetInfo

This file contains release notes for NetInfo. Items specific to later releases are listed first, and notes from previous releases follow.

## Notes Specific to Release 4.2

### Changes to NetInfo internals

The following changes were made to improve performance, reliability, and enhance system administration:

#### Performance monitoring enhancements

- Additional log messages have been provided in the **nibindd** and **netinfod** daemons. These log messages, issued via **syslog**, can be used to diagnose problems as well as to monitor the behavior of the NetInfo servers.

Note: Most of the log messages are issued at LOG\_INFO and LOG\_DEBUG priorities.

- The **syslog** facility for the **netinfod**, **nibindd**, and **lookupd** daemons are all configurable. The facility can be specified with a new *-F facility* command line option. **nibindd** passes this flag along to **netinfod** on restart. In addition, the **syslog** facility for a given domain's **netinfod** can be overridden using a new **logging\_facility** property in the **root** directory for that domain.

Note: If the -F *facility* flag is not specified on the command line and the **logging\_facility** property is not specified then the default facility of LOG\_DAEMON will be used.

- You can now specify the minimum priority of messages which are logged via **syslog**. This can enhance performance on a busy netinfo server. The **syslog** priority for a given domain's **netinfod** is specified using the **logging\_level** property in the **root** directory for that domain.

Note: If the **logging\_level** property is not specified then the default priority of LOG\_DEBUG will be used.

- In addition to the database **checksum** and the **server\_version**, the NetInfo **statistics** operation now provides the following information:

Property	Value(s)
tag	Database tag; master or clone; number of clones if master (2 or 3 values)
ip_address	Host's Internet address
hostname	Host's name
write_locked	<b>No</b> if writes allowed, <b>Yes (a readall)</b> , <b>Yes (n readalls)</b> , or <b>Yes (due to SIGINT)</b> ; <b>clone</b> if server is a clone.
notify_threads	Number currently active, maximum, update latency in seconds (3 values)
notifications_pending	Number of updates awaiting propagation
authentications	Four integers, in pairs. The first number in each pair is the number of successful authentications of the type; the second is the number of unsuccessful authentications. The first pair represents statistics for operations as a whole (e.g., an <b>ni_create</b> operation); the second, for low-level operations, including handling of, e.g., <b>_writers_key</b> .
readall_proxies	Maximum number of proxies, and whether <b>strict</b> or <b>loose</b> (2 values). Always <b>0</b> and <b>loose</b> for clones.
cleanup_wait	Number of minutes between cleanup resync periods, and number of minutes remaining in current cleanup time (2 values)
total_calls	Number of calls received by the daemon, excluding this call
binding	<b>unknown</b> if an rparent request has not yet been issued, <b>notResponding</b> if there is a <b>serves</b> property indicating a parent domain but that the specified host(s) are not responding, <b>root</b> if no parent domain is specified, <b>forcedRoot</b> if the <b>isRoot</b> property was specified, or <b>address/tag</b> of the

most recent binding.

Following the **binding** property is a sequence of properties, one for each NetInfo operation. The key for the property is the name of the operation; each property has three values: the SunRPC procedure number, the number of times the procedure has been invoked (except this one, for **statistics**), and the total amount of time spent in that routine, in  $\mu$ sec.

- The process title displayed by, e.g., **ps(1)** now shows whether the **netinfod** is a master or a clone of the domain. Readall proxies show their tag and the address of the clone receiving the database. For example:

```
157 ? SW      0:02 -master of local (netinfod)
158 ? SW      0:00 -clone of worknet (netinfod)
160 ? SW      0:00 -clone of test1 (netinfod)
161 ? SW      0:35 -master of testing (netinfod)
630 ? SW      0:00 -testing->129.18.11.216 (netinfod)
```

## Performance improvements

A number of performance improvements have been made to NetInfo to ensure that NetInfo scales well into a network with many clones spread across a large wide area network .

- **readall** processing has been improved. The master now can fork a child process to respond to a clone's **readall** request. These children are called <sup>a</sup>readall proxies<sup>o</sup>. This new behavior is configured by setting the **readall\_proxies** property in the domain's **root** directory. The property has up to two values: first, the maximum number of readall proxies; second, whether *only* a readall proxy may respond to a **readall** request (if the maximum number of proxies are running), which is called <sup>a</sup>strict<sup>o</sup> proxies. If the number of proxies allowed is **-1**, an unlimited number of proxies will be used, up to the system's resource limitations. By default, no proxies are used; by default, if proxies are used, strict proxies will be used.

One *important* side-effect of using readall proxies is a change in the way modifications are handled when a master is replying to a **readall** request. Without proxies, the master won't handle the request, and won't reply to the request, until it's done replying to the **readall**. This would lead to read requests timing out; in conjunction with preferential binding and the way the client library handles reconnecting, a master's machine could appear catatonic until the **readall**

completed because the **lookupd** on that machine is trying, continually, to contact the master. In the case of write requests, the client would, likely, time out waiting for the master to respond; the change was made, though, because it's in the master's request queue.

Rather than just hanging on the request, when readall proxies are used the master will be able to handle the request. If the request is a write of some sort, the master will reply with **NI\_MASTERBUSY**.

- In order to decrease network load, **netinfod** now uses UDP to obtain registration information from **nibindd**. If the attempt over UDP fails, TCP is used.

## Other changes and enhancements

- **nibindd** now supports **netinfod** aliasing, allowing multiple servers on the same machine potentially to respond to parent requests for the same tag. The modification allows the binder daemon to substitute some other NetInfo server (an "aliased" server) for the server that was requested by a client. For instance, if a client (NetInfo server for tag **local**) requests a given tag (**network**), then the binder daemon passes the request off to some other server (**netinfod network-33** perhaps). The criteria for deciding which requests (they're all for **network**) get passed to which server is based on some additional NetInfo properties.

The extra information resides as properties in the domain's root directory, using the **alias\_name** and **alias\_addrs** properties. The former specifies the alias to which this domain will respond; the latter is a series of values containing the address and netmask of clients which should be referred to this domain when the alias name is requested. The address and netmask are one value, separated by an ampersand (&). For example:

```
sabre% niutil -read -t mustang/network-41 /
master: mustang/network-test
alias_name: network
alias_addrs: 192.42.172.0&255.255.255.0 192.42.173.0&255.255.255.192
```

This says that any requests for tag **network** from machines with IP addresses that match either 192.42.172 in the high-order 24 bits or that match 192.42.173.0 in the high-order 26 bits should get handled as if they were asking for tag **network-test**.

- **nibindd** can now be configured to contact the **netinfod** upon receipt of a **getregister** request. Provide **nibindd** with the **-P** flag, and an argument of the number of seconds to wait before assuming the **netinfod** has crashed or is catatonic. **Warning:** the binder daemon is completely unresponsive to any other requests while it is waiting for the results of this contact attempt.
- All write operations to a domain will be disabled upon receipt of a INT signal to the corresponding **netinfod** process. Subsequent **INT** signals will toggle the read-only status.
- **netinfod** must use privileged ports for many of its operations. If the system runs out of privileged ports, operations can fail. The acquisition of a privileged port will now be retried for 10 seconds.
- PIDs for each **netinfod** process are now written to **/etc/netinfo/tag.pid**.
- A number of additional properties can be added to the **root** directory of a domain which control the behavior of the corresponding **netinfod** daemons.

**cleanup\_wait** Periodically, the master **netinfod** ensures that its clones are synchronized, and that its list of clones is current. This period can be now configured using the new **cleanup\_wait** property in the domain's root directory. The value is the number of minutes between cleanups. The default is 30 minutes. If the value is negative, no periodic resync is done on clones.

**cloneReplyReadall** Normally, only the master server of a domain will reply to a **readall** request. If the **cloneReplyReadall** property exists in the domain's **root** directory, clones will also reply to **readall** requests. This property is reexamined upon receipt of a **resync**.

**isRoot** The **netinfod** daemons for this domain will assume that they are the root of the domain hierarchy regardless of the values of the **serves** properties in the **/machines** subdirectories.

Note: If the presence of this property changes, a **resync** must be sent to *all* the **netinfod** daemons to ensure that all servers for the domain update their notion of whether they are running the root domain or not. In

addition, once a **netinfod** determines it is running the root domain, there is a thirty (30) minute timer which must expire before it re-examines this determination.

## logging\_facility

The **netinfod** daemons for this domain will use the specified facility for all messages issued via **syslog**. This property can have multiple values using one of the following formats:

hostname:facility sets the facility for the specified host  
facility default case for unlisted hosts

Here's an example of the master logging to the LOG\_LOCAL0 facility and the clone(s) logging to the LOG\_LOCAL6 facility:

```
sabre% niutil -read -t 127.0.0.1/testing /  
master: sabre/testing  
logging_facility: local0 sabre:local6
```

## logging\_level

The **netinfod** daemons for this domain will use the specified priority as the minimum level of messages issued via **syslog**. This property can have multiple values using one of the following formats:

hostname:level sets the priority level for the specified host  
level default case for unlisted hosts

Here's an example of the master logging all messages with a priority of LOG\_DEBUG and above with the clone(s) logging messages with a priority of LOG\_ERROR and above.

```
sabre% niutil -read -t 127.0.0.1/testing /  
master: sabre/testing  
logging_level: sabre:debug error
```

## sanitycheck

Some additional checks were added into Release 3.3 where a clone would check with the master's **netinfod** daemon to ensure that the correct **serves** properties were in place such that this clone would successfully

receive update propagations. Due to problems with this code it was subsequently disabled in the 3.3patch. While this problem has been corrected we opted to keep the default behavior such that the check will not be made. To enable this feature a **sanitycheck** property can be added to the **root** directory of the domain.

subthreads	The number of "notification" subthreads, threads used to propagate updates, can now be configured by adding the new <b>subthreads</b> property in the <b>root</b> directory of a domain. The default number of subthreads is 5.
update_latency	The update latency, the delay between when an update arrives at the master and when its propagation to the clone server(s) commences, can now be configured with the new <b>update_latency</b> property in the <b>root</b> directory of the domain. Lengthening the time interval allows for coalescing of updates which can be beneficial if a large number of changes are made in a short interval of time. The value of the property is the number of seconds to wait. The default is 2 seconds.

## New command-line utility behavior

- The **niutil** command supports the following new operations:

<code>-sstat</code>	print server statistics (this is a new abbreviation for the <code>-statistics</code> operation)
---------------------	---

- **niutil -rparent** now accepts a `-n` flag which will print the numeric IP address for the remote parent rather than the hostname.
- **niutil -t** won't hang on a hostname lookup when **lookupd** is hung if an address is provided instead of a name.
- We have improved the exit status codes returned from the **niutil** command. The command will exit with `100+status` when a NetInfo operation returns a non-zero status. The status codes are defined in the file `/usr/include/netinfo/ni_prot.x`.

For example: **niutil -rparent** exits with **NI\_NETROOT** (`100+11`) when querying the root

domain.

## Notes Specific to Release 4.0

### Changes to NetInfo internals

- A new version of the **lookupd** daemon was introduced in this release. Additional details can be found in the file **/NextLibrary/Documentation/NextAdmin/ReleaseNotes/lookupd.rtf**.