Table of Contents

1. NoMachine Enterprise Products, Server Administrator's Guide
   1.1. About This Guide
   1.2. Document Convention and Important Notices
   1.3. Resources on the Web

2. The NoMachine Back End
   2.1. The NoMachine Configuration Files

3. Managing Server and Services
   3.1. Accepting Connections
   3.2. Stopping and Starting Server and Services
   3.3. Stopping and Starting a Network Service
   3.4. Hiding the NoMachine Monitor Icon
   3.5. Hiding Monitor Notification Messages
   3.6. Keeping Server Status GUI Visible
   3.7. Hiding the Whiteboard and Chat Tools
   3.8. Handling with Discovering of the Server on LAN

4. Messaging Users
   4.1. Sending Messages to Connected Users
   4.2. Greeting Messages

5. Supported Connection Protocols
   5.1. Defining Protocol in Server Configuration
   5.2. Connecting by NX Protocol
   5.3. Authentication Methods Available with the NX Protocol
   5.4. Connecting by SSH Protocol
   5.5. Authentication Methods Available with the SSH Protocol
   5.6. Generating a Custom SSH Key Pair for the NoMachine Login
   5.7. Distributing the Private Key to Clients
5.8. Restoring the SSH Key Pair for the NoMachine Login

5.9. Connecting to a Server Behind a Firewall (UPnP Port Mapping)

5.10. Using NoMachine Dbs for Managing User Access

6. Users Management
   6.1. Managing Users on the Server Host
   6.2. Listing the Available Resources

7. NoMachine Sessions Management
   7.1. Monitoring Sessions
   7.2. Managing Sessions
   7.3. Setting a Virtual Desktop Environment on Linux
   7.4. Connecting as Root User
   7.5. Automatic Disconnection of Users

8. Controlling Copy and Paste Operations

9. Executing Custom Scripts Triggered on NoMachine Events
   9.1. Executing Custom Scripts on Server Events
   9.2. Executing Custom Scripts on Node Events

10. Connections to Desktops and Collaborative Sessions
   10.1. Blanking the Physical Desktop while Somebody is Connected (NoMachine 5 or later)
   10.2. Disabling Connections to the Physical Desktop
   10.3. Configuring Interaction Level to the Physical Desktop
   10.4. Requiring Authorization for Connecting to a Physical Desktop
   10.5. Disabling Connections to the Virtual Desktop on Linux
   10.6. Configuring Interaction Level to the Virtual Desktop
   10.7. Requiring Authorization to Connect to a Virtual Desktop
   10.8. Creating a list of Trusted Users to Connect to the Desktop
   10.9. Enabling the X11 Vector Graphics Mode in Virtual Desktops

11. Connect by the Web
   11.1. Disabling Access via Web
12. Sharing Devices through NoMachine
   12.1. Connecting Disks
   12.2. Disabling Disk Connection
   12.3. Connecting Printers
   12.4. Disabling Printer Connection
   12.5. Connecting USB Devices
   12.6. Disabling USB Device Connections
   12.7. Connecting Network Ports
   12.8. Disabling Network Port Connection
   12.9. Connecting Smart Card Devices

13. Disabling File Transfer
   13.1. Disabling File Transfer

14. Multimedia and Session Recording
   14.1. Supporting Audio and Microphone
   14.2. Disabling or Setting Audio Support
   14.3. Disabling Session Recording and Recording of the Local Desktop
   14.4. Setting a Video Codec and Framerate

15. Behavior of NoMachine 5 vs. NX 3.5.0
   15.1. Listing the Available Desktop Types
   15.2. Activating the Disconnect/Terminate Dialog

16. Configuring the Automatic Updates
   16.1. Setting Frequency for Automatic Updates Check

17. Logging Facilities
   17.1. Using the System Logging Facilities (NoMachine 4.4 or later)
   17.2. Retrieving Logs of Server, Network Services and Device Services
   17.3. Enabling Debug Log Levels for Server, Node and Web Player
   17.4. Retrieving Logs of Display Server
   17.5. Retrieving Logs of the Display Agent
17.6. Retrieving Logs of the Client Monitor

17.7. Redirecting Logs to a Custom File

17.8. Disabling the Logging of X Clients with Virtual Sessions on Linux

17.9. Setting the Session Log Maximum Size

17.10. Configuring the Automatic Clean-up of Session Directories

18. Miscellaneous

18.1. Supporting OpenGL Applications in Virtual Session

18.2. Setting the Number of Threads for the Display Server

18.3. Setting priority for the Display Server and Display Agent
1. NoMachine Enterprise Products, Server Administrator’s Guide

Welcome to the System Administrator’s guide for NoMachine 5 or later. Most common operations detailed in this guide can be performed by the NoMachine UI and the Server preferences panel running on the local installation of the server. How to manage the server from the graphical tools is however part of a different manual.

This document proposes instead an overview of the server functionalities and provides an aid to tune the NoMachine system behaviour to fit specific environments or needs. Instructions and commands reported in this guide are intended to be run from command line in a shell or console. When applicable, they are suitable for all the supported platforms: Linux, Mac and Windows.

NoMachine Server Functionalities
NoMachine offers a range of server products to meet different needs, which can vary from personal to business use. Basic functionalities are common to all the NoMachine products. Additionally, there is a set of advanced functionalities, such as multi-node capabilities, profiles and session redirect, which are supported only by a specific type of server. A separate guide deals with such advanced functionalities.

Tuning the NoMachine Server Behaviour
Installation is conceived to provide a fully operative NoMachine server: setup configures the software taking care of installation and launching all the necessary services. It is also possible to tune the NoMachine system further to fit very specific scenarios.

NoMachine for the Web
Install NoMachine Cloud Server, to support connections via web. This package includes a small Apache web server, nxhtd, and the web player, a web alternative to the player for running sessions. Once the installation has been successfully completed, the web player should be up and running without the need for any further configuration.

The NoMachine Server Packages
Any of the NoMachine server packages include the NoMachine GUI which provides the graphical interface (Server preferences) for administering the server and its services. This GUI acts also as client for running sessions and connecting to remote desktops.

The NoMachine Server and Node Command Line Tools
Both NoMachine server and node programs have a command line interface to execute operations. You need to be a privileged system user to access all these functionalities. These commands can be run from an xterm or similar on Linux and Mac from the command prompt (cmd.exe) on Windows.

1.1. About This Guide
This guide is organized into the following macro-sections:

**Backend**
Overview of the NoMachine backend, configuration files, and subscription files.

**Management of the NoMachine Services**
Start and stop access to the NoMachine system, enable and disable server and services.

**Authentication Management**
Access NoMachine by NX protocol, configure the NX Network Server (nxd), connect by SSH protocol (the latter via system or NoMachine login), replace the default RSA keys used for the NoMachine login, fit a custom SSH daemon configuration.

**User Management**
Create and remove system and NoMachine users, get information about users, enable and disable the ability to start a session and more.

**Session Management**
Get information about sessions, terminate sessions, tune the NoMachine server behaviour to control functionalities such as use of clipboard or persistence of virtual desktop sessions.

**Custom Scripts**
Instruct the server and/or the node to execute custom scripts triggered on server/node events.

**Collaborative Sessions**
Tune the behaviour of sessions connecting to either a physical or virtual desktop (e.g., request the owner's authorization before connecting to the desktop and the interaction level with the session).

**Resource Sharing, Printing and Other Services**
Share directories and files, printing support, forward USB devices over the network, transfer files and more.

**Multimedia and Session Recording**
Manage audio and microphone support. Configure session recording and recording of the local desktop.

**Logging Facilities**
Manage log levels and log files.
1.2. Document Convention and Important Notices

The following conventions are used in this guide:

1) **BaseDirectory** indicates the base directory where the NoMachine binaries and libraries are installed. By default, **BaseDirectory** is "/usr" on Linux, "C:\Program Files\" on Windows and "/Applications" on Mac.

2) **InstallationDirectory** is **BaseDirectory/NX** on Linux, **BaseDirectory/NoMachine** on Windows and **BaseDirectory/NoMachine.app** on Mac.

3) Samples included in this guide refer to the the default location for the installation.

**Important notices:**

1) **Server and node commands** to be executed from command line require to specify the full path. Format of commands used in the next chapter assumes implicitly that the full path is specified.

Full path to execute nxserver and nxnode tools is:

on Linux

BaseDirectory/NX/bin/nxserver  <--COMMAND>

BaseDirectory/NX/bin/nxnode  <--COMMAND>

on Mac:

BaseDirectory/NoMachine.app/Contents/Frameworks/bin/nxserver  <--COMMAND>

BaseDirectory/NoMachine.app/Contents/Frameworks/bin/nxnode  <--COMMAND>

on Windows:
BaseDirectory/NoMachine/bin/nxserver  <--COMMAND>
BaseDirectory/NoMachine/bin/nxnode  <--COMMAND>

where  <--COMMAND> is the command to be executed.

For example to retrieve the server usage on Linux, run:

/usr/NX/bin/nxserver  --help

On Linux and Mac, commands may be also invoked by specifying:

/etc/NX/nxserver  <--COMMAND>
/etc/NX/nxnode  <--COMMAND>

2) Commands and configurations require administrative privileges, even when done via the NoMachine GUI.

3) For a complete list of all server and node commands, run respectively 'nxserver  --help' and 'nxnode  --help' commands.

1.3. Resources on the Web

The NoMachine website, http://www.nomachine.com provides a variety of online resources in conjunction with the software and its usage:

The NoMachine Packages
The latest version of the NoMachine packages is available at: http://www.nomachine.com/download

The Installation Guides
For detailed information about how to install and upgrade your NoMachine installation, view the guides available at: https://www.nomachine.com/all-documents

The Knowledge Base
The NoMachine Knowledge Base, https://www.nomachine.com/support, provides technical
documents especially for administrators as well as end-user guides, articles, and “howto's.”

Leave Feedback About This Guide
Our goal is to provide comprehensive and clear documentation for all NoMachine products. If you would like to send us your comments and suggestions, you can use the contact tool available at https://www.nomachine.com/contact-request, selecting Web Quality Feedback as your option.

2. The NoMachine Back End

2.1. The NoMachine Configuration Files

The configuration files for server and node are server.cfg and node.cfg. If Cloud Server is installed, the web player also comes with a configuration file, cloud.cfg.

They are placed in the:

BaseDirectory/NX/etc directory on Linux

BaseDirectory/NoMachine/etc directory on Windows

BaseDirectory/NoMachine.app/Contents/Frameworks/etc/ directory on Mac.

Format of the Configuration Files
All the configuration files use a format without the need for specifying additional separators such as equals and quotes. Quotes are necessary only if the specified value contains multiple parameters.

The Default Configuration
Server and Node come with a default configuration that is sufficient to grant a working setup in the majority of environments. It will be up to the NoMachine administrators to tune the installation according to their specific needs by setting the related configuration keys.

Edit the Configuration Files
NoMachine configuration files are text files made up of a number of key-value pairs. All the configuration files can be edited manually by a text editor. For example ‘vi’ can be used on Linux and Mac, and 'notepad' on Windows. On Windows it can be necessary that you copy the cfg file in a different place, edit it and move it to the etc directory.
IMPORTANT

1) Uncomment the configuration key (i.e., remove the '#' pre-pended to the key) to set a value different from the default.

2) When a configuration key supports an on/off status, set value to '0' to disable it and to '1' to enable it.

Make Changes to the Default Configuration Effective
Changes will be effective with the next new session without the need of restarting the server if not otherwise specified.

3. Managing Server and Services

3.1. Accepting Connections

You can stop and start accepting new connections from the Server status GUI ('Stop the server' and 'Start the server' respectively): this will prevent users from running new connections, but will not terminate connections already running.

To disable accepting new connections from the command line, run:

nxserver --stop

To enable accepting new connections, run:

nxserver --start

3.2. Stopping and Starting Server and Services

All NoMachine services can be stopped by the Server status GUI ('Exit the server'). When doing so, you will be asked if services must be started at the next reboot or not. You can restart services also from the Server status GUI ('Run the server').

To stop all the NoMachine services from the command line, run

nxserver --shutdown
This will completely disable access to the server host machine and terminate all connections already running on that host. By default, all services will be restarted when the machine is booted. To override this behavior, specify the `--startmode` option:

```bash
nxserver --shutdown --startmode manual
```

To start server and services run:

```bash
nxserver --startup
```

All services will be restarted at the next reboot. To not start services when the machine is rebooted, specify:

```bash
nxserver --startup --startmode manual
```

It's possible to set the 'start mode' (if services will be started automatically at boot or not) by using:

```bash
nxserver --startmode manual
nxserver --startmode automatic
```

To stop and restart server and services run:

```bash
nxserver --restart
```

### 3.3. Stopping and Starting a Network Service

You can stop a single service from the Server status -> Server preferences -> Network services GUI. You can choose there also the start mode: if the service must be started automatically at the next boot or not.

To stop a service from command line, run:

```bash
nxserver --stop SERVICE
```

where SERVICE can be:

- nxd, the Network Server for accepting connection by NX protocol
- nxhtd, the HTTP server for serving web connections (available for Cloud Server only)
- nxsshd, the SSH server for Windows
To start or restart a service, run respectively:

```
nxserver --start SERVICE
```

and:

```
nxserver --restart SERVICE
```

By default each service is automatically restarted at the next boot. You can configure that on a per-service basis by running:

```
nxserver --start mode SERVICE manual
```

or

```
nxserver --start mode SERVICE automatic
```

to restore the default behavior.

These commands operate on the configuration keys listed below. You can change them manually in the server configuration.

Enable automatic starting of the NX Network server, nxd:

```
StartNXDaemon Automatic
```

Disable automatic starting of the NX Network server, nxd:

```
StartNXDaemon Manual
```

Enable automatic starting of the HTTP server, nxhtd:

```
StartHTTPDaemon Automatic
```

Disable automatic starting of the HTTP server, nxhtd:

```
StartHTTPDaemon Manual
```

Enable automatic starting of the SSH server, nxsshd:

```
StartSSHDaemon Automatic
```

Disable automatic starting of the SSH server, nxsshd:

```
StartSSHDaemon Manual
```

---

### 3.4. Hiding the Monitor Icon
3.5. Hiding Monitor Notification Messages

By default NoMachine issues a balloon message to notify about events like user’s disconnection or user's requests for connecting. You can disable notification messages by setting the following key in the node configuration:

DisplayMonitorNotifications 0

In this case when a connection request arrives, the server assumes that it's not accepted and the user will be unable to connect. To always accept connections to the physical desktop without requesting for authorization, set the following key in the server configuration:

PhysicalDesktopAuthorization 0

and to always accept connections to a virtual desktop (available on Linux only), set:

VirtualDesktopAuthorization 0

Restarting the server is necessary to make changes effective:

nxserver --restart
The Monitor icon cannot be shown when the system tray is not available. If you need to have quick access to the Server Status GUI, you may enable the following key in the node configuration:

AlwaysDisplayMonitorWindow 1

In this way the Server Status panel is always visible and cannot be closed.

Restarting the server is necessary to make changes effective:

nxserver --restart

### 3.7. Hiding the Whiteboard and Chat Tools

If you want to disable the possibility of launching the Whiteboard from the Monitor menu, edit the node configuration file to have:

EnableWhiteboard 0

Restarting the server is necessary to make changes effective:

nxserver --restart

### 3.8. Handling with Discovering of the Server on LAN

By default NoMachine server broadcast information to let other NoMachine computers to discover it on the local network. You can disable this feature from the Server status -> Server preferences -> Server options GUI or by setting the following key in the server configuration:

EnableNetworkBroadcast 0

Restarting the server is necessary to make changes effective:

nxserver --restart

### 4. Messaging Users

#### 4.1. Send Messages to Connected Users

You can send messages to a connected session by running the server commands by command line.
To send a message to all running sessions

`nxserver --broadcast "Your message goes here"`

or to send a message only to the session specified by its session id:

`nxserver --message <sessionid> "Your message goes here"`

Note that if multiple users are connected to the same desktop, they will all see the message.

### 4.2. Greeting Messages

It is possible to welcome users when session is started by issuing a greeting message.

Update the node configuration file by writing the text of your message in any of the following keys:

- `NodeFirstLoginGreeting "Welcome to your first NX session"
- `NodeLoginGreeting "Welcome back to your NX session"

Message specified in the `NodeFirstLoginGreeting` key is displayed only the first time a user runs a session there. Message specified in the `NodeLoginGreeting` key is shown each time a user starts a new session.

### 5. Supported Connection Protocols

NoMachine uses by default connection by NX protocol which is its own protocol for secure communication over the network. Encryption in the NX protocol is implemented using OpenSSL TLS/SSL, based on 2048bit RSA private/public key exchange and the ECDHE-RSA-RC4-SHA cipher suite. This suite uses elliptic curve ephemeral Diffie-Hellman keys exchange and RC4 stream cipher with 128 bit (16 bytes) keys for strong encryption. The Enterprise NoMachine products provide tunneling of connections using SSH and full integration with any authentication backend supported by the host SSH server.

When using the NX protocol, NX data can travel on TCP and UDP streams, even at the same time. The client and server can decide dynamically what transport to use, based on the type of data and the network conditions. Client and server negotiate the UDP transport at session startup, after having negotiated the main TCP link. UDP uses symmetric Blowfish encryption, with key negotiated
on the secure TCP link. UDP is presently not available when using the SSH tunneling, to ensure that all data goes through the same SSH link, as it was in old version 3.

### 5.1. Defining Protocol in Server Configuration

Protocols are defined in the ClientConnectionMethods key in the server configuration. They are specified as a comma-separated list of values:

```
ClientConnectionMethods NX,SSH,HTTP
```

This key is automatically populated during the installation of the update of the package.

**Tip:** If your server supports SSH but it still reports that SSH is not available, check the ClientConnectionMethods key and ensure that the SSH values is set. Then restart the server:

```
nxserver --restart
```

### 5.2. Connecting by NX Protocol

The default setting of NoMachine is to run connections is via the NX protocol on port 4000. On server side, the Network Server, nxd, is listening on port 4000. It's mandatory that this port must be open between client and server to allow connections by NX protocol.

You can change the listen port for nxd from the Server status -> Server preferences -> Network services GUI or by editing value of the following key in the server configuration file:

```
NXPort 4000
```

Then restart the service:

```
nxserver --restart nxd
```

When NX protocol is used, UDP communication for multimedia uses by default a range of ports between 4011 and 4999. These ports must be open between client and server. If they are not available, traffic will fall back to TCP communication. You can change port range, define a comma-separated list of ports or a single port by changing value set for the following key in the server configuration:

```
UDPPort 4011-4999
```

Users can disable UDP in their connection settings from the NoMachine GUI in the Advanced panel for the NX protocol settings.

### 5.3. Authentication Methods Available with the NX Protocol
The NX protocol supports:

- Authentication with password
- Authentication with private SSH key (key-based authentication)
- Kerberos ticket-based authentication.

Users can select the authentication method in their connection settings from the NoMachine GUI in the Advanced panel for the NX protocol settings.

### 5.4. Connecting by SSH Protocol

The default port used for the SSH protocol is 22 on Linux and Mac. On such platforms NoMachine relies on the SSH server installed on the system. If your SSHD is configured to listen on a port different from 22 you must align the server configuration accordingly. Set a proper port number for the following key in the server configuration:

```
SSHDPort 22
```

Users will have to specify such value in their connection settings.

On Windows, NoMachine provides the SSH server, nxsshd, listening on port 4022. You can change the port from the Server status -> Server preferences -> network services GUI. This setting corresponds to the value specified in the following server configuration key:

```
SSHDPort 4022
```

After changing the port value, it's necessary to restart nxsshd.

Port for the SSH server (by default 22 or 4022 in case of a server on Windows) must be open between client and server. This is mandatory to allow connections by SSH protocol.

### 5.5. Authentication Methods Available with the SSH Protocol

The available authentication methods are:

1) System login, it means that the user must have a valid account on the system. The System login via SSH supports most authentication methods supported by SSH, including:

- Password;
- Key-based;
- Key-based stored on smart card;
- Kerberos.

Additionally, it supports forwarding of key-based auth and Kerberos tickets to the node host.

2) NoMachine login, it corresponds to the authentication method supported by NX version 3.5.0. This type of authentication is limited to password based authentication. The NoMachine login allows users to authorize with a server-specif RSA key (or DSA key for older versions) or the default one if a custom key was not set on the server and their system password. Users must have a valid account on the system.

IMPORTANT:

When connecting to NX 3.5.0 servers, remember that older servers support only SSH connections and the 'NoMachine login': be sure to select the NoMachine login in the connection GUI. On the other hand, users connection by NX Client 3.5.0 will always use SSH connections and the 'NoMachine login'.

5.6. Generating a Custom SSH Key Pair for the NoMachine Login

To replace the default SSH key pair for the nx user used for authenticating with SSH and NoMachine login, please follow instructions here:

https://www.nomachine.com/AR01C00126

5.7. Distributing the Private Key to Clients

The new private key has to be distributed to all clients that have to be granted access to the server host. More detailed instructions are available here: https://www.nomachine.com/AR01C00126

5.8. Restoring the SSH Key Pair for the NoMachine Login

To restore the SSH key-pair provided with the server package you have to run from console on the server host:

nxserver --keyrestore

Then in the NoMachine (client) GUI uncheck the checkbox that reads 'Use an alternate server key' for the NoMachine login from settings for that connections.

For the web player, edit the cloud.cfg files and restore the original value for the SSHKey
configuration key.

### 5.9. Connecting to a Server Behind a Firewall (UPnP Port Mapping)

Connections via NX, SSH or HTTP protocol are possible only if NoMachine host and user's machine are on the same LAN or server has a public IP.

When the server is behind a firewall or doesn't have a public IP, you have to configure the router to forward external port to the nxd service (to use the NX connection protocol) and to the SSH server (to use the SSH protocol).

If the router on server side supports UPnP, NoMachine can try to map ports for NX and SSH. External ports are selected randomly from the 20000 - 30000 range. At session startup NoMachine will also try to map UDP ports by using UPnP. If successful, external and internal ports are the same port number.

You can enable port forwarding for the NX network server and the SSH server via the Server status -> Server preferences -> Network services gui by enabling the 'Gateway port'. This can be done also for the NoMachine HTTP server (nxhtd) which listen by default on on port 4080 and 4443 for secure HTTP connections. These ports must be open between the user's device and server. This is mandatory to allow sessions from the web.

Activating the 'Gateway port' corresponds to set the following keys in the server configuration:

- **EnableUPnP** to enable support for NAT-PMP and UPnP networking protocols
- **NXUPnPPort** to specify the port where the NX network service (nxd) will be redirected using NAT-PMP or UPnP
- **SSHDUPnPPort** to specify the port where the SSH server will be redirected using NAT-PMP or UPnP
- **HTTPUPnPPort** to specify the port where the HTTP service will be redirected

You can retrieve information about about UPnP port mapping, e.g. IP of th gateway device, external port and port mapped by running:

```
nxserver --upnpstatus
```

To terminate port mapping run:

```
nxserver ----upnpunmap
```

Note that connections from the internet may still be possible if router has been configured manually for port forwarding.
5.10. Using NoMachine Dbs for Managing User Access

Use of NoMachine dbs can be configured by editing the server configuration. The table below reports which configuration key value has to be set to enable or disable specific behavior as defined in 'Target' field:

<table>
<thead>
<tr>
<th>Target</th>
<th>Configuration Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User access based on system authentication</td>
<td>EnablePasswordDB 0</td>
<td>Authentication is requested to the system, user's connection is allowed once the user has been authenticated. The server verifies the user's password against its NX Password db. NoMachine password can be different to system password. By default all users are enabled to access the NoMachine system once authenticated. With this configuration a user can be disabled and re-enabled at any moment from command line. Every time a new account is created via NoMachine or an already existing system user runs the session for the first time, the user is added to the NoMachine NX Users DB, even when the use of NX Users DB is disabled. These users cannot be disabled.</td>
</tr>
<tr>
<td>User access based on NX Password db</td>
<td>EnablePasswordDB 1</td>
<td></td>
</tr>
<tr>
<td>Enable or disable user's access (use the NX Users db)</td>
<td>EnableUsersDB 1</td>
<td></td>
</tr>
<tr>
<td>Allow connections from all authenticated users (use the NX Users db)</td>
<td>EnableUsersDB 0</td>
<td></td>
</tr>
</tbody>
</table>

6. Users Management

6.1. Managing Users on the Server Host

You can manage (create, delete and modify) user accounts by using tools provided by your Operating System or the NoMachine server commands as explained below.

Creating a System Account

`nxserver --useradd USERNAME --system`

Creating a NoMachine Account

You need it only when authentication is based on NX Password DB. Run:
nxserver --useradd USERNAME

You can assign a password different from system password.

**Enabling and Disabling a NoMachine User**
If the server is configured to use the NoMachine NX Users DB, you can enable/disable a user. Run:

nxserver --userenable USERNAME or nxserver --userdisable USERNAME

When disabled, the user doesn't have access to NoMachine sessions.

**Retrieving the NoMachine User Authentication Type**
This allows to verify if the user's authentication is based or not on NoMachine Password db. Run:

nxserver --userauth USERNAME

**Modifying the User's Password**
The user can modify its own system password by running:

nxserver --passwd --system or, if NoMachine Password db is in use:

nxserver --passwd

You can modify user's password by running:

nxserver --passwd USERNAME --system or:

nxserver --passwd USERNAME

**Listing the NoMachine Users**
All users who have run at least one session or have been added to NoMachine dbs are stored in the Users db. You can retrieve the complete list by running:

nxserver --userlist

**Removing Account**
To remove an account from the system, you can run:

nxserver --userdel USERNAME --system

For removing the account from the NoMachine dbs:

nxserver --userdel USERNAME

**6.2. Listing the Available Resources**
When the user connects, the server inform the client about all the resources that will be available during that connections. Such resources are grouped by class: session, service, feature and node. You can see the full list by running:

nxserver --resourcelist

The availability of some resources depend only on the server type (e.g. NoMachine Enterprise Desktop allows connections to the physical desktop on Linux/Windows/Mac, i.e. it supports only a the 'physical-desktop' type, which belongs to the 'session' class. The availability of other resources depend on third party software being installed or not on the system. This is the case of audio on Linux which requires PulseAudio or Alsa to be present. When a service does not rely on external programs, it depends on the corresponding NoMachine libraries and programs. This is, for example, the case of the USB server necessary for forwarding USB devices over the network.

To know which resources are effectively available, run:

nxserver --resourcelist --value yes

This command is available also for the node, but it will list only those resources belongin to the 'session' and 'service' class:

nxnode --resourcelist --value yes

### 7. NoMachine Sessions Management

Each session on the same server is uniquely identified by a session id (which can look like: B253864E822F5A235825F3AB8853AF00) and a display id (e.g.,1002).

A session can be in any of the following statuses:

- **Connected** - when it's connected to the remote display.
- **Disconnected** - this status is available only for virtual desktop sessions. A session is marked as disconnected when it's disconnected from the remote display. A disconnected session can be reconnected at any time even from a different machine (migration). While a session is disconnected, applications on remote stay running.
- **Terminated** - the session has been closed in a clean way and all NoMachine processes have been shut-down smoothly.
- **Failed** - Any of the NoMachine processes has failed to start or it has been "un-cleanly" terminated.

Transitional statuses are "Connecting," "Disconnecting," and "Terminating."

### 7.1. Monitoring Sessions

You can monitor sessions from command line tools. Below are the server commands to be run from
Listing Running Sessions (in status Connected or Disconnected)

nxserver --list

You can also filter results on per-user basis:

nxserver --list USERNAME

Note that the number of active connections on the server corresponds to the number of sessions in status Connected.

Session History

History is preserved for a certain amount of time as set in the server configuration. To see the complete list of sessions, including those that have been cleanly terminated or failed, run:

nxserver --history

If you want to filter results on a per-user basis:

nxserver --history USERNAME

and to get more details about a session:

nxserver --history SESSIONID

Output of this last command in case of a session failed can help to understand what went wrong.

Clearing Sessions History

You can reset the history backlog by running:

nxserver --history clear

This will not clear sessions in status Connected.

Configuring the Session History Backlog

Data is preserved for 30 days. You can modify that in the server configuration file by uncommenting and setting a different value for the following key:

SessionHistory 2592000
This key accepts the following values:

< 0: Never delete data from NX session history.
0: Disable NX session history.
> 0: Keep data in session history for this amount of seconds.

### 7.2. Managing Sessions

#### Disconnecting a Virtual Desktop Session from Command Line

You can disconnect a session, if it's a virtual desktop one, by running:

```
nxserver --disconnect SESSIONID or:
nxserver --disconnect DISPLAYID
```

You can also disconnect all virtual sessions belonging to a specific user:

```
nxserver --disconnect USERNAME
```

#### Disconnecting or Terminating Sessions Automatically

If you need to disconnect a virtual desktop session or terminate a remote desktop session after a certain time of inactivity, you can specify it by uncommenting and adding the `-timeout s` (s stays for seconds) option to the DisplayAgentExtraOptions key in the NoMachine Node configuration.

For example, if you want to terminate sessions after 10 minutes of inactivity you need to set:

```
DisplayAgentExtraOptions "-timeout 600"
```

If the NoMachine display agent doesn't receive any input from the user in the given timeout, it will either disconnect or terminate the session. Termination of the session will be carried out if the session is not persistent or no X application is connected to the display. Otherwise the agent will disconnect the session so that the X applications will be left running.

#### Terminating the Session from Command Line

To terminate a virtual desktop or remote desktop session you can run:

```
nxserver --terminate SESSIONID or:
nxserver --terminate DISPLAYID
```

To terminate sessions of a certain user, run instead:

```
nxserver --terminate USERNAME
```

Finally, if you want to terminate all sessions, just restart the server:

```
nxserver --restart
```

or shutdown if you want to forbid new connections until the server is started again:

```
nxserver --shutdown
```
Terminating Automatically Virtual Desktop Sessions in Status Disconnected

It's possible to specify for how long the server has to keep alive virtual desktops in status disconnected. When the time is expired, the server will terminate virtual desktops if no user are connected there. To let the server terminate a disconnected virtual desktop after XXX seconds, edit the server configuration file, uncomment and set the timeout value (XXX) expressed in seconds in the following key:

DisconnectedSessionExpiry XXX

For example, by setting 'DisconnectedSessionExpiry 600' in the node configuration file a virtual desktop will terminate after ten minutes provided there is no activity.

Terminating Automatically Virtual Desktop Sessions when the Maximum Number of Virtual Desktops is Reached

To terminate a disconnected session when the maximum number of virtual desktops (see 'Limiting the Number of Virtual Desktops' below) is reached and make room for a new virtual desktop, enable the following key in the server.cfg file:

EnableAutokillSessions 1

Limiting the Number of Virtual Desktops

You can set a limit for the number of virtual desktops provided that such limit does not exceed the number of connections allowed by the server license value (it's the 'Virtual Desktops' field in the server.lic file).

For example, if your server allows unlimited virtual desktops and you want to allow a maximum of 10 concurrent virtual desktops, you need to edit the server configuration to have:

VirtualDesktopsLimit 10

You can also specify how many virtual desktops a single user may run. For example, to allow 2 connections per-user, uncomment and set the following key in the server configuration file:

VirtualDesktopsUserLimit 2

A Practical Example
Limit the number of virtual desktops to ten and keep alive a virtual desktop (inactive & disconnected) for one day. If a new virtual desktop is requested, the server will terminate the oldest virtual desktop in status disconnected to make room for the new session.

VirtualDesktopsLimit 10
EnableAutokillSessions 1
DisconnectedSessionExpiry 86400

**Limiting the Number of Concurrent Connections**

You can set a limit for the number of connections (either to the physical and to the virtual desktops when supported) provided that such limit does not exceed the number of connections allowed by the server license value (it's the 'Connections' field in the server.lic file).

For example, if your server allows unlimited connections and you want to allow a maximum of 10 connections, you need to edit the server configuration to have:

ConnectionsLimit 10

You can also specify how many connections a single user may run. For example, to allow 2 connections per-user, uncomment and set the following key in the server configuration file:

ConnectionsUserLimit 2

**Note for multi-node environments:**

These keys (ConnectionsLimit, ConnectionsUserLimit, VirtualDesktopsLimit and VirtualDesktopsUserLimit) are set by default to '20' in the server.cfg of the main server and of each of the remote nodes in the multi-node environment.

Custom values can be set on the server and will apply to the whole multi-node system or can be set on a specific node and will apply to this node only. Custom limits can be set also on the main server and on each of the remote nodes.

The general rule is that more restrictive settings override the others.

For example:

1) The multi-node environment is made of one server and three nodes

If ‘ConnectionsLimit 100’ is set in server.cfg on the main server and ‘ConnectionsLimit 50’ is set in server.cfg of three nodes, the maximum number of global connections distributed among the three
nodes is still 100 (and not 150 as allowed by the sum of each node setting).

2) The multi-node environment is made of one server and one node

If `ConnectionsLimit 100` is set in `server.cfg` on the main server and `ConnectionsLimit 50` is set in `server.cfg` of the remote node, the maximum number of connections on the node is 50 (and not 100). If another node is added to the system with `ConnectionsLimit 50` set in its `server.cfg`, the maximum number of global connections distributed among the two nodes is 100 (as per-settings of the main server), but each node will allow maximum 50 connections.

Note also that other limits can be set on the remote nodes by using:

i) nxserver --nodeadd NODE --limit LIMIT

The --limit option sets the maximum number of concurrent connections (defined by the LIMIT value) allowed on that node. This value is also used by the NoMachine custom script for weighted round-robin.

ii) nxserver --ruleadd --class node --type limit --value LIMIT

The limit type of rule sets the maximum number of concurrent connections (defined by the LIMIT value) allowed on the system.

iii) nxserver --ruleadd --class node --type limit --value LIMIT --node NODE

The limit type of rule sets the maximum number of concurrent connections (defined by the LIMIT value) allowed on the specific node.

The more restrictive limit always applies and overrides the others.

Disabling persistent virtual desktops

Users may be forced to terminate their virtual desktop session by setting in the server configuration:

DisablePersistentSession all

In this way when the user closes the virtual desktop, the session is terminated instead of being disconnected. This server configuration key accepts also a list of comma-separated usernames and will be applied to the specified users.

7.3. Setting a Virtual Desktop Environment on Linux
During installation, NoMachine detects the default desktop environment set on the system and configures the node accordingly. You can define an alternative desktop environment by editing the node configuration file, uncomment and set a proper value for the DefaultDesktopCommand key.

For example, if you have KDE installed on Ubuntu 12.10 and want users to be able to run new virtual KDE desktops, set the configuration key to:

```
DefaultDesktopCommand "/etc/gdm/Xsession startkde"
```

If NoMachine is configured to use the default desktop environment on the system (i.e. you didn't customize the DefaultDesktopCommand key), you can define which virtual desktop environment must be run for a certain user by configuring its `~username/.xsession` file.

### 7.4 Connecting as Root User

By default, NoMachine prevents the running of sessions as privileged (root) user. You can however configure the NoMachine Server to allow it. Do it by enabling the following server configuration key:

```
EnableAdministratorLogin 1
```

### 7.5 Automatic Disconnection of Users

By default, when all the available connections are already in use and a user is requesting to run a new connection, the server retrieves the oldest connection and requests authorization to the owner for making room for the incoming user. You can configure the server to automatically disconnect the oldest connected user to make room for the connecting user. To do that, enable the following server configuration key:

```
AutomaticDisconnection 1
```

The automatic disconnection applies when the maximum number of available connections to the desktops or the maximum number of available virtual desktops is exceeded.

### 8. Controlling Copy and Paste Operations

By default users can copy and paste from locale to the session and vice-versa.

You can configure the server to limit such operations by setting proper values in the configuration file as explained below.

**Limiting copy & paste operations**

To forbid copy & paste partially or totally, uncomment and set a proper value for the EnableClipboard key.

Possible settings are:
EnableClipboard client
Content copied on the user's side can be pasted inside the session

EnableClipboard server
Content copied inside the session can be pasted on the user's side

EnableClipboard none
No copy and paste operations are allowed

EnableClipboard both
Two-ways copy and paste operations are allowed

**Limiting the Clipboard Buffer**
By default, the clipboard buffer is unlimited. If you want, for example, to limit the clipboard buffer to 4MB, you have to uncomment and set the following key (value is expressed in bytes) in the NoMachine Node configuration file:

```
ClipboardBufferLimit 4194304
```

### 9. Executing Custom Scripts Triggered on NoMachine Events

#### 9.1. Executing Custom Scripts on Server Events

The server configuration provides a number of keys that can be activated to execute a custom script upon a certain event. According to the event, a number of parameters can be specified for each script. Events related to session disconnections and reconnection (e.g., the UserScriptBeforeSessionDisconnect key) apply only to virtual desktop sessions.

Custom scripts are executed on the NoMachine server and are common to all the users who are accessing the server. Since the server is running as the nx user, you have to grant this user the necessary permissions in order to execute the custom script.

For example, if you want to run a script to log the remote IP of the connecting user, you need to ensure that the output can be written in a directory on the server host accessible by the nx user.

Note that if the execution of the scripts fails, the server will terminate. You can override this behavior by forcing exit 0 inside the custom script.

**KEY**
```
UserScriptBeforeLogin ""
UserScriptAfterLogin ""
UserScriptBeforeSessionStart ""
UserScriptAfterSessionStart ""
UserScriptBeforeSessionDisconnect ""
UserScriptAfterSessionDisconnect ""
UserScriptBeforeSessionClose ""
UserScriptAfterSessionClose ""
UserScriptBeforeSessionReconnect ""
```

**PARAMETERS**
```
remote ip
username
session id, username, node host, node port, main session id(*), main session type(*)
```
UserScriptAfterSessionReconnect ""
UserScriptBeforeSessionFailure ""
UserScriptAfterSessionFailure ""
UserScriptBeforeCreateUser ""
UserScriptAfterCreateUser ""
UserScriptBeforeDeleteUser ""
UserScriptAfterDeleteUser ""
UserScriptBeforeEnableUser ""
UserScriptAfterEnableUser ""
UserScriptBeforeDisableUser ""
UserScriptAfterDisableUser ""

(*) 'main session id' and 'main session type' parameters are available only when the user connects to an already running virtual desktop (session shadowing).

NOTES FOR MULTI-NODE ENVIRONMENTS

1) Custom scripts related to login events have to be placed in the server.cfg file on the main server host.

2) Custom scripts related to session events (included session failure) have to be placed in the server.cfg file on each of the remote nodes.

3) Custom scripts related to user events (create, delete, enable, disable) have to be placed in the server.cfg file on the main server host.

9.2. Executing Custom Scripts on Node Events

The following keys (available in the node configuration file) allow to execute a custom script on a certain NoMachine node event. According to the event, a number of parameters can be specified for each script. Also in this case events related to session disconnection and reconnection are available only with virtual desktop sessions (e.g., the UserScriptBeforeSessionDisconnect key).

Note that if the execution of the scripts fails, the node will terminate. You can override this behavior by forcing exit 0 inside the custom script.

KEY
UserScriptBeforeSessionStart ""
UserScriptAfterSessionStart ""
UserScriptBeforeSessionDisconnect ""
UserScriptAfterSessionDisconnect “”  
UserScriptBeforeSessionClose “”  
UserScriptAfterSessionClose “”  
UserScriptBeforeSessionReconnect “”  
UserScriptAfterSessionReconnect “”  
UserScriptAfterSessionFailure “”

(session id, username, session type, display)

UserScriptBeforeSessionClose “”  
UserScriptAfterSessionClose “”  
UserScriptBeforeSessionReconnect “”  
UserScriptAfterSessionReconnect “”  
UserScriptAfterSessionFailure “”

(session id, username, session type, main session id(*), main session type(*))

(*) 'main session id' and 'main session type' parameters are available only when the user connects to an already running virtual desktop (session shadowing).

**NOTE FOR MULTI-NODE ENVIRONMENTS**

1) Custom scripts have to be placed in the node.cfg file on each of the remote nodes.

Some examples to use custom scripts triggered on node events are available here:

https://www.nomachine.com/AR02L00787

**10. Connections to Desktops and Collaborative Sessions**

You can configure how users will connect to a desktop owned by another user from the Server status -> Server preferences -> Desktop access GUI. You can basically determine if users can connect or not without asking the desktop owner's permission and if users will be able to interact with the desktop.

Disabling the request for desktop owner's authorization before connecting can be useful in case of remote administration of headless machines.

Allowing connections in interactive mode grants the user full access to the desktop resources and applications. View-only mode is suggested for example when making presentations or teaching a lesson.

Besides using the graphical tools, you can configure the server by editing the server configuration file, uncommenting and setting a proper value for keys as illustrated in the following paragraphs.

Configurations made from the GUI apply to connections to physical and virtual desktops. If you want to set a separate configuration for these desktops, you have to edit the server configuration manually.

Starting from version 5, the server supports the screen blanking feature: when active, the local user will see a black screen on the physical monitor while somebody is connected from remote.
Operations made on the physical screen are not shown and the local user cannot interact with the desktop until the remote user logs-out. Control is given back to the local user once the remote user has logged off.

A tutorial is available here: https://www.nomachine.com/blanking-the-host-screen-during-remote-control

10.1. Blanking the Physical Desktop while Somebody is Connected (NoMachine 5 or later)

To enable the screen blanking feature, edit the server configuration file (server.cfg), uncomment and set the following key:

EnableScreenLock 1

To disable it, set EnableScreenLock 0.

You can activate the screen blanking feature on the machine also from the Server preferences GUI panel.

**NOTE for Mac hosts:**

When the blanking on the screen is enabled on Mac, mouse will be locked but keyboard lock will be not available.

**NOTE for multi-node environments:**

Enabling or disabling the screen blanking has to be configured on each of the remote nodes by editing the corresponding server.cfg.

**NOTE for virtual hosts**

Screen blanking is available for physical hosts, it is not supported on virtual machines since it has effect on the physical monitor.

10.2. Disabling Connections to the Physical Desktop

To forbid connections to the physical desktop set the following key in the server configuration:

PhysicalDesktopSharing 0

If the server supports only connections to physical desktop, users will no longer be able to access that machine.

10.3. Configuring Interaction Level to the Physical Desktop
To forbid users to interact with the desktop once connected set in the server configuration:

PhysicalDesktopMode 0

In this way, the connected user will access the physical desktop in view-only mode.

To allow interaction instead, set:

PhysicalDesktopMode 1

10.4. Requiring Authorization for Connecting to a Physical Desktop

To request for the explicit authorization of the desktop owner before connecting the user be sure that the following key is set in the server configuration:

PhysicalDesktopAuthorization 1

To allow users connecting to the physical desktop without explicit permissions, set:

PhysicalDesktopAuthorization 0

10.5. Disabling Connections to the Virtual Desktop on Linux

By default users can connect to virtual desktop sessions. To forbid this capability, set in the server configuration file:

VirtualDesktopSharing 0

This setting disables also the listing of other user’s virtual session in the client GUI.

10.6. Configuring Interaction Level to the Virtual Desktop

To forbid users to interact with the desktop once connected set in the server configuration:

VirtualDesktopMode 0

In this way, the connected user will access the virtual desktop in view-only mode.

To allow interaction instead, set:

VirtualDesktopMode 1

10.7. Requiring Authorization to Connect to a Virtual Desktop
To request for the explicit authorization of the desktop owner before connecting the user be sure that the following key is set in the server configuration:

VirtualDesktopAuthorization 1

To allow users connecting to the virtual desktop without explicit permissions, set:

VirtualDesktopAuthorization 0

10.8. Creating a list of Trusted Users to Connect to the Desktop

For versions higher than 5.1.19, rather than allow all users to connect without desktop's owner authorization or click accept for every single user which would like to connect, it is possible to define in advance a group of trusted users who don't need the specific owner's permission.

In order to create a list of trusted users, administrators should use the nxserver commands for creating and editing users and groups. These commands provide a new option, --trusted, to define if the user is trusted for connections to the physical or the virtual desktop or both.

The general format of these commands is:

1) To create a new trusted user on the system

nxserver --useradd <username> --system --trusted [virtual | physical]

E.g.

- Create a system account for user 'nxtest01' and let this user connect to another user's desktop without the need for the owner's approval:

  nxserver --useradd nxtest01 --system --trusted

- Create a system account for user 'nxtest02' and let this user connect to physical desktops (but not virtual desktops) without the need for the owner's approval:

  nxserver --useradd nxtest01 --system --trusted physical

2) To make an existing user trusted, modify trusted permissions or remove them:

nxserver --useredit <username> --trusted [virtual | physical | none]

E.g.

- Edit user 'nxtest01' and limit his trusted authorization to virtual desktops only:

  nxserver --useredit nxtest01 --trusted virtual
- Remove trusted authorization for user 'nxtest02':

nxserver --useredit nxtest02 --trusted none

3) To list only trusted users:

nxserver --userlist --trusted

4) To create a trusted group of users

nxserver --groupadd <groupname> --trusted [virtual | physical]

E.g.

- Create users' group 'testers' and let all users belonging to this group to connect to another user's desktop without the need for the owner's approval:

nxserver --groupadd testers --trusted

- Create users' group 'developers' and let all users belonging to this group to connect to another user's virtual desktop without the need for the owner's approval:

nxserver --groupadd developers --trusted virtual

5) To make an existing group trusted, modify trusted permissions or remove them:

nxserver --groupedit <groupname> --trusted [virtual | physical | none]

E.g.

- Modify trusted authorization for group 'developers' and give access to both physical and virtual desktops without the need for the owner's approval:

nxserver --groupedit developers --trusted

- Remove trusted authorization for group 'testers':

nxserver --groupedit testers --trusted none

6) To show list of groups and trusted attribute:
nxserver --grouplist

### 10.9. Enabling the X11 Virtual Graphics Mode in Virtual Desktops

The X11 vector graphics mode (previously known as “lightweight” mode), is available in all products providing Linux virtual desktops. It works in the same way as NX 3.5.0, which was by using the X11 protocol. You can enable it to avoid problem of blurred fonts or to increase responsiveness on slow link and end-users' clients without hardware accelerated video encoding/decoding capabilities.

You can enable the X11 vector graphics mode from the Server preferences GUI -> Performances panel, or by editing the node configuration file. In that case, you will need to uncomment and enable the AgentLightweightMode key as follows:

AgentX11VectorGraphics 1

or AgentLightweightMode 1 for previous versions of the software.

**Notes:**
- If you have a multi-node setup, remember to enable the lightweight mode on each of the remote nodes.
- Starting from version 4.2 the lightweight mode is enabled by default.

### 11. Connect by the Web

Install the NoMachine Cloud Server package to provide support for connection via web through the web player application, it also provides a small web server (nxhtd) and configures the installation accordingly. The web player is up and running after the installation at the following URL:

http://serverHost:4080

or:

https://serverHost:4443

serverHost is either the IP or hostname of the machine where the NoMachine package has been installed.

*Note that the installation comes with a self-signed SSL certificate intended to be a sample. This doesn't grant a secure connection. In order to secure your NoMachine Web Player, please replace the SSL Certificate File and the SSL Certificate Key File with your own.*

These files are respectively installationDirectory/etc/keys/host/ht_host_rsa_key.crt and installationDirectory/etc/keys/host/ht_host_rsa_key.
11.1. Disabling Access via Web

To forbid access to the system via web, set the following key in the web player configuration file (cloud.cfg):

EnableWebPlayer 0

In this way, the NoMachine HTTP server (nxhtd) is forbidden to serve content of the NoMachine web player application.

12. Device Sharing

NoMachine implements a self-contained infrastructure for making available physical and logical devices over the network from local to remote or vice-versa.

The NoMachine infrastructure for device sharing ensures that all services work out of the box without the need for any additional change or configuration. It is possible to connect disks, printers, USB devices, network port and smartcards.

Connecting devices is supported only by connections run from player and can be accessed from the NoMachine toolbar within the session. Connected devices can be disconnected during the life of the session and reconnected later. If option 'Export this deviceName at session startup' is checked, this device is automatically reconnected at the next session start-up.

Besides using the server settings UI to enable or disable services, you can configure them by editing the corresponding keys in the node configuration file. This file maintains also a number of obsolete keys to preserve compatibility with older client versions (e.g., NX Client 3.5.0).

Supported services are:

**Connect a disk**
Local and remote disks can be connected and disconnected during the life of the session and navigated by file browsing. A disk connected as 'Public' is available to all users accessing that desktop. A private disk is available only to the user who connected it.

Administrators can configure paths on the server where public and private disks will be mounted as well as specifying which disks on the server can be made available to users.

This service relies on the NoMachine File-system Server (nxfsd) and NoMachine File-system Adapter driver on Windows. On Mac it uses the nxfuse extension whilst on Linux it uses fuse, installed on the system by default. The nxfs and nxfsserver programs are used on all systems to mount disks.

**Connect a printer**
Local and remote printers can be connected at any time (bi-directional printing). A connected printer is listed among the available printers when printing a document or similar. A printer can be...
connected to be ‘Public’, i.e. available to all users connected to that desktop, or private, for a specific user. It can be also configured to be the default printer.

This service relies on the NoMachine Device Server service and the NoMachine Printer Adapter driver on Windows. On Mac and Linux it uses the CUPS infrastructure present on the system. In this last case, a printer can be exported to the server only if the connected user is in the lpadmin group.

**Connect a USB device**
USB devices such as disks, pendrives, webcam etc... are forwarded through the network. For example, when a USB device is forwarded from locale (where the player is running) to remote it becomes available on the remote side only.

This service is based only on the NoMachine USB Server (nxusbd) and drivers (NoMachine USB Hub, NoMachine USB Adapter and NoMachine USB Host Adapter on Windows, nxusb.ko kernel module for Linux and nxusb.kext for Mac) and doesn't require external tools.

**Connect a network port**
It allows users to make available service ports (such as Samba, CUPS, FTP, SSH, telnet and others) from local to remote and vice-versa via a virtual network interface.

This service uses the NoMachine Device Server and the NoMachine VPN Adapter driver on Windows and Mac. On Linux it relies on a NoMachine tool plus a standard driver.

**Connect smartcard devices**
This permits forwarding of a smartcard reader from client to server side and makes smartcard authentication available within the session. The server host must support authentication via smartcard.

Support for authentication with smart card has been set-up by relying on the Public Key Infrastructure (PKI) and using an OpenSC compatible smart card. It can be integrated with Kerberos ticket authentication and ticket forwarding.

### 12.1. Connecting Disks

NoMachine allows access to local and remote file systems from within the session through the SSHFS file-sharing protocol and by means of FUSE, a technology to implement a fully functional filesystem in a userspace program.

Connected folders and disks can be disconnected during the life of the session or left as they are.

By default, all disks from the server are available to be connected to the end-user's machine. However you can specify a set of disks and folders by editing a proper value for the DiskSharingList key in the node configuration file. The default value is: all. Alternatively, you can specify a list of comma-separated directories. Note that $(HOME) and $(USER) are accepted values.

For example, on mac you might want to specify:
DiskSharingList $(HOME),/Volumes/TimeMachine

Disks from the end-user's machine can be connected on the server in 'Public' or 'Private' mode.

**Connecting public disks**
By default public disks are exported from player to "$(PUBLIC)" directory on the server, where $(PUBLIC) is:

/Volumes on Mac

/media on Linux

C:\Users\Public on Windows Vista/7/8/8.1/10

%ALLUSERSPROFILE% on XP

You can specify a different path by un-commenting and editing the DiskSharingPublicBasePath key in the node configuration file.

Note that $(USER) is an accepted value that can be also concatenated to specify the path to a directory, for example "/tmp/$(USER)".

The target directory must exist on the system.

**Connecting private disks**
By default, private disks are exported on the server to $(DESKTOP) directory. You can specify a different path by uncommenting and editing the DiskSharingPrivateBasePath.

Note that $(HOME) and $(USER) are accepted values and can be concatenated to specify path to a directory, for example "/$(HOME)/Shares".

The target directory must exist on the system.

### 12.2. Disabling Disk Connection

To forbid disk and filesystem sharing, uncomment and set a proper value for the EnableDiskSharing key in the node configuration file:
client Filesystem on the client can be connected to server side and accessed from the session.

server Filesystem on the server can be connected to the end-user's machine and accessed through the whole life of the session.

both Client and server filesystem can be connected to remote and local sides respectively.

none Neither client or server filesystem can be connected.

For example, to forbid connecting disks from remote to local side, set:

EnableDiskSharing client

12.3. Connecting Printers

The printers sharing infrastructure integrates client-side printers with the server-side printing subsystem and vice-versa. Printers available on the client machine can be shared and used within the session as well as printers on the server side which can be made available on the end-user's machine.

Connected printers can be disconnected during the life of the session or left as they are. In this case, they are automatically shared at the next session start-up.

12.4. Disabling Printer Connection

To forbid printers sharing it is necessary to uncomment and set a proper value for the EnablePrinterSharing key in the node configuration file:

client Printers on the client can be connected to server side and made available within the session.

server Printers on the server can be connected to the end-user's machine.

both Client and server printers can be connected to remote and local sides respectively.

none Neither client or server printers can be connected.

For example, to forbid a server-side printer to be connected to the end-user machine, set:

EnablePrinterSharing client

12.5. Connecting USB Devices

This service creates a USB tunnel between client and server to forward devices over the network such as hard disk, web cams, barcode readers, and pen drives from local to remote desktops and vice-versa.
12.6. Disabling USB Device Connections

To forbid USB device sharing it is necessary to uncomment and set a proper value for the EnableUSBSharing key in the node configuration file:

client
USB devices on the client can be forwarded to server side and made available within the session.

server
USB devices on the server can be connected to the end-user's machine.

both
Client and server USB devices can be connected to remote and local sides respectively.

none
Neither client or server USB devices can be connected.

For example, to avoid that users can forward a USB devices from the server to its own machine, set:

EnableUSBSharing client

12.7. Connecting Network Ports

NoMachine can create virtual network interfaces and establish a bridge between local and remote sides or vice-versa to provide transparent access to network resources.

This service allows access to any of the default network servers like Samba, CUPS, FTP, SSH and Telnet or any other type, for example a MySQL server.

Connecting a Samba server allows access to resources on that server host via the SMB/CIFS protocol. Connecting a local CUPS server to the remote side allows mounting of printers (local to the user) on that remote CUPS subsystem so that files can be printed on the remote side via the IPP protocol.

Some typical examples of usage:

Print to remote printers from the session
If you have a Linux or Mac machine you can add the locale CUPS server via the player toolbar. Choose to add a local server and select CUPS. In this way all printers that are available on your side will be available also on the server and you can print all your documents via the native CUPS (IPP) protocol.

Access a remote host not in your Network Neighborhood
If the remote host has a Samba server, you can add it via the player toolbar. Choose to add a remote server and select Samba as server type. Once that Samba server is added, the remote host shows up in your local Network Neighborhood. You can then connect to remote folders via SMB/CIFS protocol as if that host was in your local network.

Make available a client side HTTP server
You can add your local HTTP server via the player toolbar and make it available on the remote host where your session is running. In this way you can develop and test your web application directly inside the session, without the need for sharing or moving files from remote to local.

**Connect to MySQL server behind a firewall**
You can choose to add a remote server via the player toolbar. Select 'Custom' and specify MySQL and the port for the MySQL server, by default 3306. Once done, you can connect to that MySQL server via the MySQL client installed on your PC.

### 12.8. Disabling Network Port Connection
To forbid network server sharing it is necessary you uncomment and set a proper value for the EnableNetworkSharing key in the node configuration file:

- **client**: Network servers on client side can be connected and made available within the session.
- **server**: Network server on the server side can be connected and made available on the end-user's machine.
- **both**: Network servers from client and server side can be connected to remote and local sides respectively.
- **none**: Neither client or server side network servers can be connected.

For example, to forbid users from connecting their local ports to the server, set:

```
EnableNetworkSharing server
```

### 12.9. Connecting Smart Card Devices
When the smartcard reader plugged into the enduser’s host is forwarded to the server host, the smartcard authentication is made available inside the session. It can be integrated on with Kerberos Ticket system for example for implementing single sign-on (SSO).
You can enable or disable support for smartcard forwarding by uncommenting and setting the following server configuration key to 1 or 0 respectively.

For example, to disable it set in node configuration file:

```
EnableSmartcardSharing 0
```

### 13. File Transfer
When user is connected to the desktop, they have the possibility to transfer files by using the Connection Monitor tool from the system tray. The user can transfer a file from their own PC to the
remote host where the session is running and vice-versa. The same user can also send a file to a specific user connected to (sharing) the desktop, or to all connected users.

### 13.1. Disabling File Transfer

To forbid file transfer you have to uncomment and set a proper value for the `EnableFileTransfer` key in the node configuration file:

- `client`: Files can be transferred from client machine to the server.
- `server`: Files can be sent from the server to clients.
- `both`: Client and server files can be transferred on remote and locale respectively.
- `none`: Neither client or server files can be transferred.

For example, to forbid users from transferring a file from the server to their PC:

```plaintext
EnableFileTransfer client
```

### 13.2. Limiting File Transfer

Since version 4.1 it's possible to limit the transfer of files in upload and download separately. The following keys must be uncommented and set in the node configuration file. The first key activates or de-activates the ability of forbidding a file transfer when its size exceeds the limit set in the second key. Key pair for limiting files in upload are, where the size limit is in B, by default 100MB:

```plaintext
EnableUploadSizeLimit 1
UploadSizeLimit 104857600
```

and for limiting files in download are:

```plaintext
EnableDownloadSizeLimit 1
DownloadSizeLimit 104857600
```

Versions prior to 4.1. were using only 2 keys that were ruling both files in upload and in download:

```plaintext
EnableFileTransferSizeLimit 0
FileTransferSizeLimit 104857600
```

### 14. Multimedia and Session Recording
14.1. Supporting Audio and Microphone

On Linux, NoMachine audio framework is integrated with PulseAudio sound server. If PulseAudio is not available on the system, NoMachine is able to use ALSA (Advanced Linux Sound Architecture). This is automatically managed by the NoMachine server so that multimedia support can work out of the box without the need for any configuration. If both PulseAudio and Alsa are available, the administrator might want to configure the node to use one or the other.

On Windows 8, 7 and Vista NoMachine's audio system perfectly integrates with the Windows media framework. On XP instead, NoMachine relies on its own driver, the NoMachine Audio Adapter. For microphone support NoMachine always uses the NoMachine Microphone Adapter driver.

On Mac, NoMachine installs and uses its own virtual drivers to support audio and microphone seamlessly.

Audio and microphone support is available only with sessions run by the player.

Compatibility with the audio infrastructure of NX Client and Server 3.5.0 is preserved. In the case of connections from clients earlier to version 4, the NoMachine server will try to use PulseAudio in (Enlightened Sound Daemon) compatibility mode.

14.2. Disabling or Setting Audio Support

To disable audio and microphone support, uncomment and set the AudioInterface key to ‘disabled’ in the node configuration file:

AudioInterface disabled

On Linux it is possible to define whether PulseAudio Server or ALSA has to be used by setting AudioInterface key to ‘pulseaudio’ or ‘alsa’ respectively. For example:

AudioInterface pulseaudio

Accepted values on Windows and Mac are ‘nxaudio’ or ‘disabled’.

14.3. Disabling Session Recording and Recording of the Local Desktop

You can deactivate the possibility for users to record activities in their session and save them into a file on their own pc. To do that, uncomment and disable the EnableSessionRecording key in the node configuration file as follows:

EnableSessionRecording 0

To disable the recording of the local desktop instead, set the following in the node configuration file:

EnableLocalRecording 0
14.4. Setting a Video Codec and Framerate

NoMachine supports VP8, mjpeg and H.264 (if available on the system) video codecs for encoding display updates. By default it relies on its internal mechanism for choosing the most suitable codec and framerate depending on network conditions and hardware capabilities.

You might want to configure NoMachine to use a specific codec and framerate, for example to be sure to limit the use of CPU to the minimum. To do that, edit the node configuration file, uncomment and set a value for the following configuration keys depending on your needs.

**Use a specific video codec**
Specify the codec to be used. Accepted values are: VP8, H264 and MJPEG. For example, to use the less-CPU intensive codec MJPEG, set:

```
DisplayServerVideoCodec mjpeg
```

Accepted values are: VP8, F264 and MJPEG.

**Use a specific framerate**
Activate the following key:

```
DisplayServerUseVideoFrameRate 1
```

and specify the framerate to be used. Accepted values are 25, 30, 40 and 60. A lower number of frames per seconds requires a lower CPU usage to the detriment of video smoothness. To set the lowest framerate:

```
DisplayServerVideoFrameRate 25
```

15. Behavior of NoMachine 5 vs. NX 3.5.0

Starting from version 4.1, some behaviors typical of NX 3.5.0 server have been introduced:

1) A new virtual desktop is created for each new connection. The virtual desktop type (GNOME, KDE etc ...) must be specified in the connection settings: to do that in client 5 or 4, choose the desktop type the first time you run the session and remember to save the connection settings.

2) If you have a virtual desktop session already running or disconnected, you will be able to reconnect it automatically if you have only 1, or by choosing it if you have more than one.

3) You can migrate a virtual desktop session from one PC to another one: the session is disconnected and reconnected on the new side.

Such behaviors are automatically available when connecting from client 4.1, regardless of the server being version 4.0 or 4.1. Client 4.1 or later overrides values set in the ConnectPolicy key in the server configuration.
15.1. Listing the Available Desktop Types

To provide users with the list of all the available desktop types on that host (e.g., GNOME and KDE), enable the desktop option in the server configuration:

ConnectPolicy autocreate=1,autoconnect=1,automigrate=1,desktop=1,dialog=0

15.2. Activating the Disconnect/Terminate Dialog

Starting from NoMachine version 4.4.12 it's possible to configure the server for displaying a dialog to let the user decide if disconnecting or terminating the virtual desktop session (running in lightweight mode) when clicking on the X button to close the session window.

This allow administrators to re-introduce the disconnect/terminate dialog typical of NX 3.5.0 but it will override the possibility of disconnecting the session by clicking on the X window button. It will be still possible to terminate the session by executing log-out from the system menu.

The Disconnect/Terminate dialog is available also for custom session since version 5.1.7 (see: https://www.nomachine.com/FR01N03006)

For versions higher than 5.1.9, the Disconnect/Terminate dialog is available also for virtual desktops not in lightweight mode (see: https://www.nomachine.com/FR09M02966)

To enable the Disconnect/Terminate dialog, enable the 'dialog' option (i.e. set 'dialog=1') in the following key available in the server configuration:

ConnectPolicy autocreate=1,autoconnect=1,automigrate=1,desktop=0,dialog=1

16. Configuring the Automatic Updates

Starting from version 4.6.3 all server installations, included NoMachine (free), have the automatic check for updates enabled while version 5 or later supports also the automatic download of software updates made in accordance with the automatic fetch of NoMachine repositories.

This means that the server will periodically check our repositories (by default every two days) to verify if updates are available and will prompt a dialog informing the user that a new version is available but it will never automatically update the current installation. Also the download in background of a new software version will not lead to an automatic update of the current
installation.

A separate guide deals specifically with all the possible options for the automatic software updates:

https://www.nomachine.com/DT07M00085

16.1. Setting Frequency for Automatic Updates Check

NoMachine is configured to check for updates on the repository every two days. To change frequency, set a different value (in seconds) for the following key in the server configuration:

```
UpdateFrequency 172800
```

Set:

```
UpdateFrequency 0
```

to disable the automatic check for updates.

17. Logging Facilities

17.1. Using the System Logging Facilities (NoMachine 4.4 or later)

By default server, node and web player logs to the file defined in the SystemLogFile in their configuration files (server.cfg, node.cfg and cloud.cfg respectively.

It's possible to configure these applications to logs to the system log file instead.

Edit the server.cfg, node.cfg (and in case cloud.cfg), uncomment and set:

```
EnableSyslogSupport 1
```

It's necessary to restart the server and all services to make the change effective:

```
nxserver --restart
```

17.2. Retrieving Logs of Server, Network Services and Device Services

Logs of server (by default) and services on server side are in:
There are different types of logs:

- Logs of network service programs used to provide access to that host: nxd for connections by the NX protocol, nxhtd for web connections and nxsshd for connections by SSH protocol (on Windows only). Log files are respectively nxd.log, nxhtd-access.log, nxhtd-error.log and nxsshd.log.

  **Tip:** Check these logs when users cannot authenticate either by player or web player.

- Logs of server, node and web player programs: they track flow of operations executed by these programs from the starting of the session until its termination. Events are logged by default in the nxserver.log unless these applications are configured to log to system log.

  **Tip:** Check these logs when a session cannot be started or it was terminated unexpectedly.

- Logs of the Display Server and Agents.

  **Tip:** Check these logs when a session cannot be started or it was terminated unexpectedly.

- Logs of device services programs, for example nxfs.log, nxfsserver.log on Windows or nxusbd.log.

  **Tip:** Check these logs if the list of devices to be connected displayed by the player toolbar is empty.

- Installation and uninstallation logs and logs from automatic updates.

  **Tip:** Check these logs if you experienced unexpected behavior during any of the previous operations.

### 17.3. Enabling Debug Log Levels for Server, Node and Web Player.

You can enable the debug log level by uncommenting the SessionLogLevel key and set it to value 7 in the node and server configuration files as well as in the web player configuration file (cloud.cfg):

```
SessionLogLevel 7
```

To preserve logs written in the session directory once the session is terminated, edit the node configuration file and set:
SessionLogClean 0

17.4. Retrieving Logs of Display Server

Logs of the Display Server, i.e. the program in charge of sending images and events from the physical desktop to the connected session, are in:

- BaseDirectory/NX/var/log/node on Linux
- /Library/Application Support/NoMachine/var/log/node on Mac
- %PROGRAMDATA%\NoMachine\var\log\node on Windows Vista, 7, 8, 8.1 and 10
- Documents and Settings\All Users\NoMachine\var\log\node on Windows XP

Each time the Display Server starts, it creates a session directory named according to the following format:

C-hostname-display-sessionID

For example:

C-nomachine.local-1001-6809B3A854C28C3CC5AB3DFCE483E21F

When the Display Server terminates, the session directory is renamed as T-* . If for some reason it terminates in an unclean way, the directory is renamed as F-C-*.

Tip: Check these logs if the client shows 'no available sessions' when connecting to the server.

Note for NoMachine 4.6.4 or earlier:

The Display Server log directory was named 'nxnode'.

17.5. Retrieving Logs of the Display Agent

When a user connects to the server, a Display Agent is connected for that user to the Display Server. Its logs are placed in the same node directory as the display Server logs. Also in this case the Display Agent creates a session directory named as:

C-hostname-display-sessionID

For example:

C-nomachine.local-1003-32E5EA48B7D5553A3EAEAD9559043CCC
If the session is terminated correctly, the directory is deleted or renamed as T-* if the node configuration is set to preserve old session directories.

If the session terminates abruptly, the directory is renamed as F-C-*.

**Tip:** Check these logs when, for example, your session terminates unexpectedly.

*Note for NoMachine 4.5.0 or earlier:*

The Display Agent logs where saved in the user's home/.nx directory.

### 17.6. Retrieving Logs of the Client Monitor

Logs of the client monitor (the nxclient --monitor process instantiated for each session) are saved in the M-hostname-* directory in the user's home/.nx directory.

If the session is terminated, the directory is deleted or renamed as T-* if the node configuration is set to preserve old session directories.

### 17.7. Redirecting Logs to a Custom File

You can redirect logs of server, node and web player to a custom file by uncommenting and setting the path to that file in the following configuration key available in their configuration file:

```
SystemLogFile /tmp/NX.log
```

The NoMachine application must be able to write to that file also if it is running as the logged user, as it is with the node program.

### 17.8. Disabling the Logging of X Clients with Virtual Sessions on Linux

When virtual sessions are being run, logs of X clients are redirected to the user's home/.nx/SessionDirectory/clients file. In order to disable the logging of X Clients, you have to uncomment and disable the following key in the node configuration file:

```
ClientLog 0
```

### 17.9. Setting the Session Log Maximum Size

You can set the maximum size allowed for the session log file by uncommenting and setting a value (in bytes) for the SessionLogLimit key in the node configuration file. For example, to set the maximum size to 4 MB:

```
SessionLogLimit 4194304
```
When the maximum size is exceeded, the node terminates the session.

When virtual sessions are running, you can limit also size of X clients log files. For example, to set the maximum size at 4 MB:

ClientLogLimit 4194304

17.10. Configuring the Automatic Clean-up of Session Directories

In its default configuration, the NoMachine server removes the session directory from the user's home/.nx directory once the session has been correctly terminated.

You can preserve it by uncommenting and disabling the following key in the server configuration file. In this case, the session directory in user's home/.nx is renamed as T-C*:

SessionLogClean 0

18. Miscellaneous

18.1. Supporting OpenGL Applications in Virtual Session

NoMachine provides VirtualGL libraries that can be loaded when running a virtual session on Linux. This allows OpenGL applications, namely 3D applications, to use server side graphics hardware.

In order to activate support for VirtualGL, do the following:

 Applies to all OpenGL applications

1. Run the following script

   /usr/NX/scripts/vgl/vglsrv_config -config +s +t +f

2. Uncomment and set the EnableVirtualGLSupport key in /usr/NX/etc/node.cfg:

   EnableVirtualGLSupport 1

3. Reboot the machine or restart the X server selected to provide HW acceleration to OpenGL rendering.

OR
Applies to a specific OpenGL application

1. Run the following script

```
/usr/NX/scripts/vgl/vglserver_config -config +s +t +f
```

2. Uncomment and set the EnableVirtualGLSupport key in /usr/NX/etc/node.cfg:

```
EnableVirtualGLSupport 1
```

3. Reboot the machine or restart the X server selected to provide HW acceleration to OpenGL rendering.

4. Then launch the OpenGL application by running inside the NoMachine session:

```
/usr/NX/scripts/vgl/vglrun <opengl app>
```

**IMPORTANT:**

If you have a multi-node environment, to use VirtualGL on all the nodes of a multi-node server, the same configuration will have to be applied to each node of the server.

In the future, the only step required will be to set the EnableVirtualGLSupport key in /usr/NX/etc/node.cfg to 1.

### 18.2. Setting the Number of Threads for the Display Server

When video is being streamed, the Display Server (in charge of providing image updates and event changes to the connected Display Agents) uses as many threads as the number of CPU cores available on the host machine.

You can specify the number of threads to be used by editing the following key in the node configuration file. Its value must be an integer number and cannot be higher than the number of physical CPUs on that machine.

For example, if you have a quad-core machine usually used for heavy operations, you might want to run only 2 threads for video streaming on 2 dedicated CPUs and leave 2 cores free for other operations. In this case, change value from 'auto' to '2' as below:

```
DisplayServerThreads 2
```

### 18.3. Setting priority for the Display Server and Display Agent

NoMachine automatically sets the process execution priority for the Display Server and Display
Agent depending on the platform and on the case. You can specify a fixed priority by uncommenting and setting a value for the following keys in the node configuration file:

DisplayServerPriority
DisplayAgentPriority

Accepted vaules are:

**realtime**
process has niceness -20 set on Linux and Mac and real time priority class on Windows. This is the most favorable scheduling.

**high**
process has niceness -10 set on Linux and Mac and high priority class on Windows.

**normal**
process has niceness 0 set on Linux and Mac and normal priority class on Windows.

**low**
process has niceness 19 on Linux and Mac and idle priority on Windows. This is the least favorable scheduling