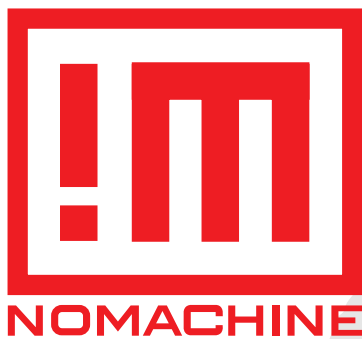


# Einführung in die NX-Technologie



# Index

<b>1. Einführung in die NX-Technologie</b> .....	<b>3</b>
1.1 Die Bedeutung offener Standards für Network Computing .....	3
1.2 Ein „Mission Statement“ zu Open Source .....	3
1.3 Weshalb das X-Window-System? .....	4
1.4 Die Anpassung des X-Window-Systems an gestiegenen Ressourcenbedarf .....	5
1.5 Die Komprimierungstechnologie von NoMachine .....	5
1.6 Die Server- und Desktop-Plattform Linux .....	6
1.7 Die Einsetzbarkeit mit RDP und RFB-Protokoll .....	7
1.8 NX als Ausgangspunkt für ein Peer-to-Peer-Netzwerk .....	7
1.9 Aufteilung der Arbeitslast auf die Knoten .....	8
1.10 Auf einfache Anwendung ausgerichtet .....	9

# 1. Einführung in die NX-Technologie

## Grundlegendes zu Merkmalen und Entwicklungsleitlinien der NX-Technologie

Das vorliegende Dokument beschreibt die Hintergründe und konzeptionellen Ansätze, die für die NX-Entwicklung maßgeblich waren. Es zeigt, wie sich NX von vergleichbaren Technologien unterscheidet und gibt einen Ausblick auf die weiteren Ziele des NX-Projekts.

### 1.1 Die Bedeutung offener Standards für Network Computing

Der Begriff „Network Computing“ steht für die Vision, dass Anwender vollkommen unabhängig von einem bestimmten PC und dessen Betriebssystem arbeiten können. Die für die Datenverarbeitung benötigten Anwendungen stehen dabei auf Servern zum Abruf bereit und können dadurch flexibel genutzt werden.

Die Geschichte des Internets hat gezeigt, dass allgemein anerkannte Protokolle (wie HTML und XML) für den Erfolg eines solchen dezentralen Modells entscheidend sind. Die mangelnde Durchsetzung vergleichbarer offener Standards im Bereich Network Computing erschwert dessen Implementierung und laufende Betreuung, insbesondere beim Einsatz im großen Maßstab. Dies dürfte einer der wesentlichen Gründe dafür sein, dass Network Computing bislang vergleichsweise schwach verbreitet ist. NoMachine setzt mit dem Rückgriff auf das weit verbreitete X-Window-System und der Freigabe wesentlicher Komponenten als Open Source an dieser Stelle an.

### 1.2 Ein „Mission Statement“ zu Open Source

NoMachine hat alle Kernbibliotheken und Low-Level-Softwarekomponenten, die für die Unterstützung der NX Distributed Computing Architecture entwickelt wurden, als Open Source freigegeben. Damit will NoMachine einer möglichst großen Entwickler-Community Gelegenheit geben, gemeinsam interaktive Server- und Client-Software zu realisieren, dank derer Network Computing zu einer Selbstverständlichkeit wird.

#### NoMachine ist GPL

Basiskomponenten sind Open Source und unter der GNU Generic Public License freigegeben

Zu den freigegeben Komponenten gehören die X-Agenten, der Client-Proxy und alle Bibliotheken, die den komprimierten Transport des X-Protokolls implementieren. NoMachine hat sich dabei für das Modell der GNU General Public License entschieden und setzt somit auf die Lizenzvereinbarung, die auch die Antriebsfeder bei der Entwicklung des Betriebssystems Linux ist.

### 1.3 Weshalb das X-Window-System?

Die NX Distributed Computing Architecture kombiniert diverse Open-Source-Technologien und handelsübliche Tools, um Network Computing so einfach und selbstverständlich zu machen wie das Surfen im Web. Zur Architektur gehört eine auf das Wesentliche reduzierte Schicht von Serversoftware, die es jedem Unix-Rechner ermöglicht, als Terminal-Server zu fungieren. Auf der Client-Seite kann eine breite Palette von Plattformen und Betriebssystemen eingebunden werden. Als Grundlage der von NoMachine entwickelten Lösung dient das bekannte und weit verbreitete X-Window-System.

#### X-Window-Ansatz

Das X-Protokoll ist von Anfang an für Network Computing konzipiert.

Das X-Window-System ist ein offenes und erweiterbares Client-Server-Protokoll, auf dem die grafischen Benutzeroberflächen der Betriebssysteme Linux und Unix basieren. Es trennt zwischen der auf den Servern ausgeführten Anwendungslogik und der auf Clients stattfindenden Darstellung. Damit folgt schon der grundlegende konzeptionelle Ansatz von X-Window dem Gedanken des Network Computing.

Vergleicht man andere Network-Computing-Lösungen mit X-Window, so wirken diese dagegen eher wie Tools, die nur behelfsweise für den Einsatz in diesem Bereich gedacht sind. Die Anwender können damit zwar auf ihrem Desktop-Computer arbeiten, doch es scheint, als ob die verantwortlichen Entwickler diese Systeme nur als Notlösung für diesen Zweck angepasst haben.

Dies ist beispielsweise das Problem bei VNC und RDP. Beide Protokolle sind theoretisch hervorragend für schlanke Clients geeignet, da sie wesentlich einfacher sind als X. Diese Einfachheit bezahlt der Nutzer aber mit unzureichenden Funktionen und mangelnder Effizienz. Dies zeigt sich zum Beispiel bei der Darstellung des Remote-Bildschirms, für die diese Protokolle riesige Mengen von Bilddaten über das Netzwerk übertragen müssen. RDP ist zwar anspruchsvoller und effizienter als andere Protokolle (z.B. RFB). Dennoch bleibt erkennbar, dass es nur als Zusatz des zugrunde liegenden Betriebssystems konzipiert wurde, statt auf die konstante Nutzung von Rechnerressourcen optimiert zu sein.

Während diese Protokolle lediglich eine Erweiterung des grafischen Subsystems des Host-Betriebssystems darstellen, arbeitet X-Window selbst als grafisches Subsystem. Da die X-Anwendungen über das X-Protokoll direkt mit X-Window kommunizieren, läuft die Übertragung wesentlich einfacher ab. Das Betriebssystem kann darauf verzichten, eine spezielle Schicht für die Übersetzung von Bildschirmaktualisierungen in ein Netzwerkprotokoll vorzuhalten.

#### 1.4 Die Anpassung des X-Window-Systems an gestiegenen Ressourcenbedarf

Wie die bereits angesprochene Trennung von Anwendungslogik und clientseitiger Darstellung zeigt, wurde X-Window ursprünglich für Network Computing konzipiert. Dabei setzt das native X-Protokoll allerdings Netzwerke mit hoher Bandbreite und niedrigen Latenzzeiten voraus, um optimal zu arbeiten. Der durch die Weiterentwicklung der grafischen Benutzeroberflächen gestiegene Bedarf an Netzwerk- und Rechnerressourcen stellt deswegen für X-Window eine Herausforderung dar, der mit neuen Mitteln zu begegnen ist.

Eine Lösungsmöglichkeit hätte darin bestanden, die X-Anwendungen auf unterschiedliche Schichten zu verteilen, um Netzwerkressourcen flexibel dort zu nutzen, wo sie gerade zur Verfügung stehen. Statt dessen gingen die X-Client- und X-Toolkit-Entwickler davon aus, dass die Nutzer auf die Netzwerkfunktionen von X-Window verzichten und nur noch die auf ihren Rechner installierte Anwendungen verwenden würden. Dadurch hat das X-System in den letzten Jahren seine ursprüngliche Rolle als Netzwerkprotokoll vollständig eingebüßt. Selbst informierte Anwender mit ausgeprägtem technischen Wissen sehen X-Window derzeit nur als einfachen Display-Treiber.

Eine derart einschneidende Beschränkung der Einsatzmöglichkeiten lässt sich mit einer effizienten Komprimierungstechnologie durchbrechen. Diese Technologie zu entwickeln war von Anfang an das Hauptziel des NX-Projekts. Durch intelligente Komprimierung sollte es jedem Anwender möglich sein, die Originalversionen der gängigsten X-Desktop-Umgebungen über jede beliebige Netzwerkverbindung auf einem Standard-X-Server auszuführen. Damit sind die Voraussetzungen geschaffen, um X-Window mit all seinen Vorzügen wieder als leistungsfähiges Netzwerkprotokoll zu nutzen.

#### 1.5 Die Komprimierungstechnologie von NoMachine

##### Kompression und Performance

Durch Caching-Methoden und Reduktion des Protokoll Overheads wird die Leistung erhöht

NoMachine hat spezielle X-Protokoll-Komprimierungsverfahren und integrierte Proxy-Agenten entwickelt, mit denen komplette Remote-Desktop-Sitzungen über Schmalband-Internet-Verbindungen möglich sind. Die NX-Technologie setzt dazu an unterschiedlichen Stellen an und kombiniert die positiven Effekte. So können sogar Verbindungen über

langsame 9600-Baud-Modems unterstützt werden. Die NX-Komprimierung arbeitet dabei auf drei Ebenen des X-Protokolls:

- NX komprimiert mit diversen, für jede Nachricht spezifischen Maßnahmen den eigentlichen Netzwerkverkehr. Dazu gehören u.a. Differential-Algorithmen, fortschrittliche Caching-Methoden sowie verlustfreie und/oder verlustbehaftete Bildkomprimierung.
- NX reduziert die Zahl der Round-Trips im Netzwerk fast auf Null und maximiert dadurch den Durchsatz.
- NX passt die Bandbreite in Echtzeit an die Netzwerkbedingungen an.

Abhängig von der gerade angezeigten Anwendung ermöglicht NX damit eine Komprimierung des X-Protokolls von 10:1 bis 100:1 und mehr. Da hierbei die Leistung in schnellen LANs nicht beeinträchtigt wird, können Remote-Sitzungen in der Regel mit einer Geschwindigkeit ausgeführt werden, die nicht von der des lokalen Rechners zu unterscheiden ist. Auch für langsame Verbindungen ist eine Effizienz gewährleistet, die selbst die Ausführung kompletter Remote-Desktop-Sitzungen im Vollbildmodus erlaubt.

### 1.6 Die Server- und Desktop-Plattform Linux

NX-Serverprodukte arbeiten mit dem Betriebssystem Linux. Linux gehört zu den Unix-Betriebssystemen. Es erfüllt damit die in modernen Unternehmensumgebungen vorausgesetzten Ansprüche an Zuverlässigkeit, Leistung und Skalierbarkeit.

Aufgrund seiner Flexibilität ist Linux gleichzeitig hervorragend geeignet, um das verteilte Peer-to-Peer-Netzwerk der Zukunft zu betreiben. Im Network Computing der nächsten Generation werden Millionen angebundener Rechner ihre Dienste und Anwendungen in einem einzigen großen Netzwerk bereitstellen. Linux bietet schon heute die Voraussetzungen, um diese Vision wahr werden zu lassen.

Im Markt der Serverbetriebssysteme ist Linux schon länger etabliert. Inzwischen hält Linux dank Desktop-Umgebungen wie KDE oder GNOME und dem preisgekrönten Office-Paket StarOffice auch im PC-Bereich mehr und mehr Einzug. Alle drei genannten Umgebungen/Produkte arbeiten wie NX mit dem X-Window-Protokoll. NX bietet eine kostenlose Unterstützung dieser Umgebungen mit deutlichen Vorteilen bei der Gesamtperformance.

### 1.7 Die Einsetzbarkeit mit RDP und RFB-Protokoll

Die Funktionalitäten von NX für das Network Computing können außer in den genannten Linux-Umgebungen (Desktops und Server) auch unter zahlreichen anderen Betriebssystemen genutzt werden. NX übersetzt dazu die Protokolle dieser Umgebungen in das X-Protokoll. Dies geschieht zum Beispiel mit dem Remote Desktop Protocol (RDP), das bei der Microsoft Windows NT/2000 Terminal Server Edition und Citrix Metaframe zum Einsatz kommt. Die gleiche Möglichkeit besteht für das Protokoll Remote Frame Buffer RFB. RFB unterstützt das ebenfalls für die Fernsteuerung gedachte Open-Source-Programm VNC.

#### Protokollübergreifend arbeiten

RFB und RDP werden in X-Protokoll übersetzt und übertragen

RDP- und RFB-Sitzungen können auf diesem Weg um einen Faktor 2 bis Faktor 10 komprimiert werden. Darüber hinaus bietet die NX-Unterstützung von RDP und RFB den Vorteil, dass nahezu jeder Rechner mit einem System erreichbar wird. So kann der Anwender über NX ganz einfach eine einheitliche Sicht auf alle für ihn über das Internet erreichbaren Applikationen erlangen.

### 1.8 NX als Ausgangspunkt für ein Peer-to-Peer-Netzwerk

Bei der Entwicklung der NX Distributed Computing Architecture stand von Anfang an eine optimierte Lastverteilung auf Knoten in einem WAN im Vordergrund. Fernziel ist die Realisierung eines Szenarios, in dem NX-Server anspruchsvolle Management-Funktionen über das NX-Netzwerk ausführen. Die Server wären dann beispielsweise für die Authentifizierung und Aktivierung von Anwendersitzungen zuständig und würden gewährleisten, dass die Knoten den Anwendern ortsunabhängig deren vertraute Rechnerumgebung (das heißt z.B. das gewohnte Dateisystem und die üblichen Anwendungen) zur Verfügung stellen.

Die Sitzungen würden auf Rechnern mit NX-Knoten in einem virtuellen Cluster ausgeführt. Dazu könnten mehrere hundert NX-Knoten an einen oder mehrere Steuerungsserver angeschlossen werden und mehrere tausend Sitzungen gleichzeitig unterstützen.

NX-Knoten wurden nicht als dedizierte Rechner konzipiert, obwohl es wahrscheinlich ist, dass sich dies in Legacy-Umgebungen zu einem gängigen Szenario entwickelt. Statt dessen könnte jeder Rechner ein Knoten eines verteilten NX-Netzwerks sein. Damit würde ein NX-Netzwerk stark den Peer-to-Peer-Netzwerken ähneln, in denen Clients sonst z.B. nach Musikdateien gesucht haben. Mit NX würden die Clients jetzt in ähnlicher Form nach einer Anwendung oder einer Desktop-Umgebung suchen. Die Such-

funktionalität in diesem Netzwerk würde dann einerseits den Client zum geeigneten Server lenken und andererseits dem Server den Nachweis der Zugriffsberechtigung des Clients liefern, auf Basis dessen der Knoten die Verbindung akzeptiert.

### 1.9 Aufteilung der Arbeitslast auf die Knoten

#### Skalierbarkeit

durch Einsatz mehrerer Knoten mit Lastverteilung

In einem derart gestalteten Netzwerk wird es den Servern möglich sein, die Netzwerklast gleichmäßig zu verteilen, indem sie Sitzungen mehreren unterschiedlichen Knoten zuweisen. Gleichzeitig könnten die Knoten die in einer Sitzung ablaufenden Anwendungen über verschiedene Anwendungsserver ausführen lassen. Dies ist zum Beispiel dann denkbar, wenn eine Ressource oder Anwendung auf dem ursprünglichen Knoten gerade nicht verfügbar ist. Performancevorteile können in einer Sitzung auch dadurch erschlossen werden, dass fallweise auf die Ressource zugegriffen wird, die „näher“ beim Anwender und deswegen - dank des kürzeren Netzwerkpfades - effizienter ansprechbar ist.

#### Eine Sitzung lässt sich in diesem Zusammenhang wie folgt beschreiben:

Die Sitzung entspricht einem Agenten, der auf einem Knoten ausgeführt wird. Dabei stehen sowohl für das X-Protokoll als auch für die Netzwerkprotokolle RDP und RFB Agenten zur Verfügung. Alle Agenten werden in der Nähe von Anwendungsservern ausgeführt. Sie akzeptieren die Verbindungen von den NX-Clients. Als Clients werden in diesem Fall die Rechner der Anwender bezeichnet, auf denen der X-Server ausgeführt wird. Dabei wird je Agent nur eine Verbindung zum X-Server benötigt.

Je nach Protokoll ergeben sich an dieser Stelle unterschiedliche Möglichkeiten. Für die Protokolle RDP und RFB sind diese relativ schlicht. Diese Protokolle emulieren einen kompletten Bildschirm. Sie verlangen keinerlei Kommunikation zwischen verschiedenen, mit demselben X-Server verbundenen Anwendungen. Ein Knoten kann jede beliebige RDP- oder RFB-Sitzung mit einem anderen Windows-Terminal-Server-Rechner oder VNC-Server verbinden. Damit wird die Last der Sitzungen gleichmäßig auf die verfügbaren Netzwerkressourcen verteilt.

Mit dem X-Protokoll sind dagegen zwei Betriebsarten möglich, je nachdem, ob der Anwender für die Sitzung den bestehenden oder einen neuen virtuellen Desktop wählt.

Wenn ein Anwender eine Sitzung auf einem neuen virtuellen Desktop ausführt, fungiert der Agent selbst als virtueller X-Server. Er fasst dabei mehrere X-Client-Verbindungen zu einer Verbindung zusammen, die zum realen X-Server führt. Die X-Anwendungen können so auf verschiedenen Knoten betrieben werden, die als Anwendungsserver dienen. Die

Kommunikation zwischen X-Anwendungen wird lokal auf der Seite des Agenten gelöst. Diese zum Teil sehr ressourcenintensive Kommunikation läuft also ohne Beanspruchung des Netzwerks ab. Die Gesamtleistung wird dadurch erheblich verbessert.

Zur weiteren Optimierung wird im übrigen gerade ein „rootless“ Agent entwickelt. Bisher erscheinen X-Anwendungen, die innerhalb eines X-Agenten ausgeführt werden, noch verschachtelt in dessen Fenster. Das heißt die Anwendung wird auf dem lokalen Desktop als Fenster im Fenster des virtuellen Desktops dargestellt. In Zukunft soll der Agent ohne ein eigenes Fenster auskommen und damit dieser Verschachtelung ein Ende bereiten.

Unproblematisch ist die Situation schon jetzt bei der zweiten Möglichkeit, d.h. wenn ein Anwender Anwendungen in der aktuellen X-Sitzung ausführen lässt. Die Fenster „schweben“ dann auf dem Desktop und integrieren sich perfekt mit den lokalen Anwendungen. In diesem Fall fungiert NX als transparenter Proxy, der den X-Verkehr komprimiert und die zusätzlichen Netzwerkdienste (wie etwa Multimediadienste oder das SMB-Protokoll) „tunnelt“. Wirkt NX in dieser Form als Proxy statt als Agent, entfällt allerdings die Auflösung der lokalen Round-Trips des X-Protokolls.

### 1.10 Auf einfache Anwendung ausgerichtet

Obwohl die NX-Technologie in einer solchen Beschreibung sehr komplex wirkt, ist sie in der Anwendung sehr einfach. Dafür sorgt unter anderem die integrierte Netzwerkfunktionalität von Unix. Auch die gute Skalierbarkeit von X-Window und den Unix-Netzwerkdiensten trägt dazu bei.

Gleiches gilt für die Wahrung grundlegender Sicherheitsaspekte. NX verwendet die Remotefunktionen von SSH für den Zugriff auf die Funktionen eines Knotens. Die gesamte Kommunikation wird mit SSL verschlüsselt, bedient sich also des am weitesten verbreiteten Public-Key-Verfahrens. Die Knoten kommen so ganz ohne neue Netzwerkservers aus. Benötigt wird lediglich ein SSH-Daemon, also ein Server, der standardmäßiger Bestandteil jeder modernen Unix-Distribution ist.

Auch im Sicherheitsbereich bestätigt sich somit die grundsätzliche Ausrichtung von NX auf einfache aber effiziente Lösungen.

#### Sicherheit durch SSH

NX nutzt SSH als sichere Übertragungstechnik

# Leistungsspektrum der Millenux GmbH

Millenux ist Premium Partner von NoMachine

## Als Premium Partner von NoMachine bietet Millenux folgende Leistungen

- Beratung
- Entwicklung
- Implementierung
- Support

## Das Millenux-Portfolio für Linux im Enterprise-Umfeld umfasst außerdem

- Open-Source-Beratung
- System-Integration
- Linux-Migration
- System-Programmierung

## Kontakt

### Millenux GmbH

Lilienthalstr. 2/1  
70825 Stuttgart-Korntal  
Germany

Tel.: +49.711.88770.300  
Fax: +49.711.88770.349  
Mail: [info@millenux.com](mailto:info@millenux.com)  
[www.millenux.com](http://www.millenux.com)