This document was written for GUUG FFG 2002, URL: http://www.feyrer.de/IPv6/guug2002-6to4.html

Rapid Deployment von IPv6 mit 6to4

Hubert Feyrer <hubert@feyrer.de>, 30. Januar 2002

Connectivity & Wechsel zu IPv6

Wir wollen uns damit befassen, wie man IPv6 Connectivity erhält, und - nachdem dies nicht allzu einfach ist - im Detail auf Altnernativen in Form von Tunnel-Lösungen eingehen, die noch einige Zeit benutzt werden, bis echte IPv6-Uplinks verfügbar sind.

Findet man einen Internet Provider der selbst IPv6 anbietet, so kann man sich glücklich schätzen. Was man als nächstes braucht ist ein Router, der die Daten bewältigen kann. Bislang bieten jedoch nicht alle Router-Hersteller Unterstützung für IPv6 für ihre Geräte an. Und selbst falls man einen Hersteller findet der IPv6 unterstützt, dann ist es unwahrscheinlich dass Routing bzw. Switching durch IPv6-fähige Hardware beschleunigt werden, was dazu führt dass ein Gerät, das IPv4 bei einer bestimmten Bandbreite (2MBd, 34MBd, ...) bearbeiten kann dieses noch lange auch nicht mit IPv6 können wird. Eine Alternative zur den heute verbreiteten Routerhardware ist die Verwendung eines als Router konfigurierten PCs, z.B. indem man Linux oder ein BSD-basierendes Betriebssystem wie z. B. NetBSD verwendet, und Software wie "Zebra" zum Implementieren der Routing-Protokolle verwendet. So konfigurierte Router sind nicht unüblich in Bereichen, in denen heute IPv6 Connectivity benötigt wird. Die Nachteile sind dass ein ISP benötigt wird, der IPv6 anbietet, sowie ein eigener, für IPv6 reservierter und dadurch kostspieliger Uplink.

Falls diese Option ausscheidet, so kann man durch Verwendung von Tunnels dennoch IPv6 Connectivity erhalten. Anstatt direkt IPv6 "on wire" zu sprechen werden die IPv6 Packete in IPv4 Pakete eingepackt. Anschliessend werden die so eingepackten Pakete über die bestehende IPv4-Infrastruktur an eine Gegenstelle gesandt die neben IPv4 auch IPv6 spricht und die die eingepackten IPv6 Pakete wieder auspackt, und über IPv6 weiterleitet.

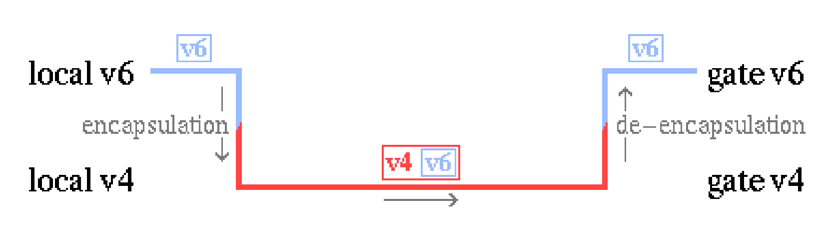

Abb.: Eine häufig genutzte Methode beim Übergang von IPv4 auf IPv6 ist das Tunneln von IPv6 in IPv4 Paketen.

Benutzt man Tunnel, so ergeben sich zwei Möglichkeiten. Zum einen sog. "konfigurierte" Tunnel, und "automatische" Tunnel. Das Aufsetzen eines konfigurierten Tunnels ist üblicherweise mit

Vorbereitungen auf beiden Seiten des Tunnels verbunden, oft in Form einer Registrierung bei der Informationen für den Setup ausgetauscht werden. Ein Beispiel für konfigurierte Tunnel ist der in [RFC1933] beschriebene IPv6-über-IPv4 Mechanismus, wie er z.B. im "gif" Treiber der auf KAME basierenden BSD-Stacks zu finden ist.

Ein automatischer Tunnel besteht aus einem Server der z.B. über das 6Bone über IPv6 Connectivity verfügt. Die Konfigurationsdaten dieses Servers sind frei zugänglich, und er hat ein Tunnel-Protokoll laufen, das keine vorherige Registrierung und Konfiguration von Rechnern oder Netzen erfordert, die ihn als Uplink benutzen wollen. Beispiel für einen weit verbreiteten automatischen Tunnel ist der in [RFC3056] beschriebene 6to4 Mechanismus wie er z.B. in "sit" Treiber bei Linux oder im "stf" Treiber bei den BSD-Stacks zu finden ist. Ein weiterer automatischer Tunnel-Mechanismus der ebenfalls keine vorherige Registrierung bei einem Uplink erfordert ist "6over4". Hierbei wird ein physikalisches Broadcast-Medium wie z.B. Ethernet oder FDDI durch eine auf IPv4-basierende Multicast-Gruppe ersetzt. 6over4 ist in [RFC2529] beschrieben. Der grösste Nachteil dabei besteht in der vorausgesetzten IPv4-basierten Multicast-Struktur. Ist diese nicht vorhanden, so ist zum Aufsetzen ebensoviel Aufwand an Verwaltung und Konfiguration nötig wie für einen konfigurierten IPv6 Tunnel, so dass sich dieser Ansatz üblicherweise nicht lohnt.

6to4 - Aufsetzen und loslegen!

Nachdem wir nun einen groben Überblick über die verschiedenen Arten, IPv6 Connectivity zu erhalten gegeben haben wollen wir uns nun im Detail dem 6to4 Mechanismus widmen. Mit dem bisher gegebenen Hintergrundinformationen sollte 6to4 nicht schwer aufzusetzen zu sein. Konfigurationen werden am Beispiel von RedHat Linux 7.0 mit dem USAGI IPv6 Stack und dem auf dem KAME IPv6-Stack basierenden NetBSD 1.5 gezeigt.

Mittels 6to4 ist es einfach, IPv6 Connectivity für Rechner zu erhalten, die nur über einen Zugang zum Internet über IPv4 verfügen. Das Protokoll kann dabei sowohl mit fest als auch mit dynamisch vergebenen IPv4-Adressen benutzt werden, wie sie heute bei DSL- und Modem-Dialup Setups üblich sind. Es bleibt anzumerken dass ein Ändern der IP-Adresse z.B. durch erneutes Einwählen ein Problem für eingebende Verbindungen ergibt, und es ist dadurch nicht ohne weiteres möglich, Server-Dienste wie FTP oder WWW anzubieten. Die Probleme bei IPv6 und 6to4 sind hier die selben wie bei IPv4.

Wo erhält man Adressraum?

Bei 6to4 erhält man selbst keine einzelne IPv6 Nummer, sondern gleich ein ganzes /48 Netzwerk! Die IPv6-Adressen werden hierbei von der eigenen (einzelnen) IPv4-Adresse abgeleitet. Der 16 Bit Adress-Präfix "2002::" ist für 6to4 basierende Adressen, d.h. für IPv6 Adressen die von IPv4 Adressen abgeleitet sind, reserviert. Die nächsten 32 Bit nach dem Präfix bestehen aus der eigenen IPv4 Adresse. Daraus ergibt sich ein /48 Netzwerk, das man für seine eigenen Zwecke z.B. zuhause oder im Büro nutzen kann. Es bleiben - wie bereits besprochen - 16 Bits für 2^16 IPv6 Subnetze, von denen jedes wiederum bis zu 2^64 Rechner aufnehmen kann. Durch den 6to4 Präfix und die eigene weltweit eindeutige IPv4 Adresse ist dieser Netzblock weltweit eindeutig, und dem Benutzer der v4-Adresse zugewiesen. Die Umrechnung der eigenen IP-Adresse nach Hexadezimal kann bequem mit dem printf(1) Unix-Befehl bewerkstelligt werden.

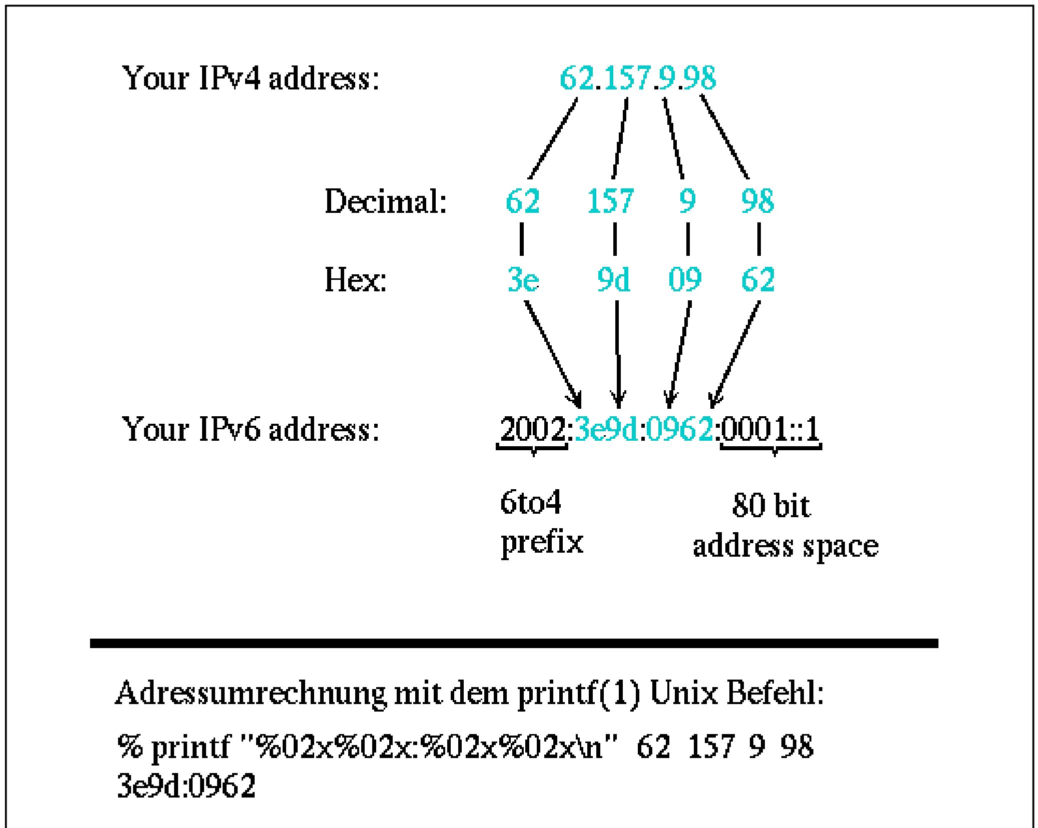

Abb.: Genug Platz für Experimente und Produktivbetrieb - 6to4 leitet IPv6 Adressen von IPv4 Adressen ab; Unix: Adressumrechnung mit printf(1)

Anbindung über entfernte Tunnelendpunkte

Bei 6to4 werden die eigenen IPv6-Pakete in IPv4 eingepackt und über IPv4 an einen ans 6Bone angeschlossenen Relay router weitergeleitet. Im Gegensatz zum konfigurierten IPv6-über-IPv4 Tunnel muss man sich bei 6to4 jedoch nicht vorher bei diesem Router registrieren. Beim Senden werden die IPv6-Pakete mittels eines 6to4-fähigen Netzwerk-Interfaces in IPv4-Pakete eingepackt und über die bestehende IPv4-Infrastruktur versandt. Beim empfangenden Relay Router der als Uplink fungiert wird das IPv6 Paket dann wieder ausgepackt, und über die dort vorhandene IPv6-Infrastruktur, z. B. eine Anbindung an den 6Bone, weitergeleitet.

Nachdem die eigene IPv4-Adresse in der Absenderadresse der versandten IPv6-Pakete enthalten ist kann jeder beliebige Relay Router den eigenen Rechner bei Antworten direkt adressieren. Dies führt dazu, dass Antworten nicht unbedingt - oder gar eher selten - von dem Relay Router empfangen werden, an den die ausgehenden Pakete gesandt werden. Im lokalen Router wird das empfangene Packet dann durch ein 6to4-sprechendes Netzwerk-Interface ausgepackt, und gemäss den Routing-Informationen über IPv6 weitergeleitet. Damit können dann auch mehrere Rechner über eine IPv4-Adresse mit IPv6-Connectivity versorgt werden.

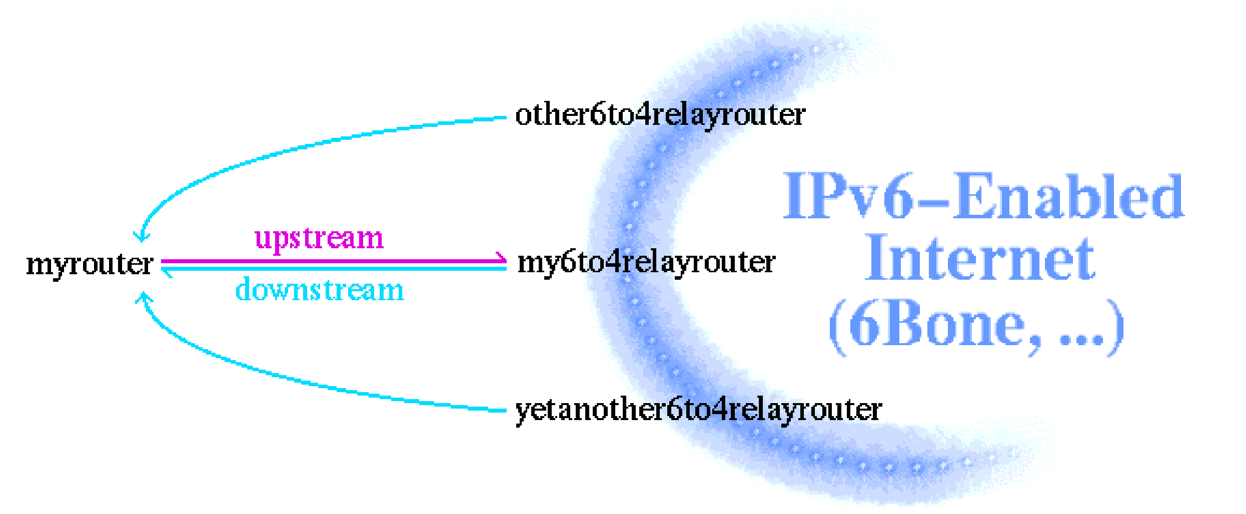

Abb.: Anfrage und Antwort können bei 6to4 über unterschiedliche 6to4 Gateways laufen

[RFC 3056] definiert die folgenden Begriffe:

- *Host*: spricht native IPv6 mit Router, Adresszuweitung über Router Advertizement vom Routers
- *Router*: kapselt IPv6-Pakete in IPv4, und sendet diese über IPv4 an einen ans 6Bone angeschlossenen Relay Router
- *Relay Router*: Empfängt gekapselte 6to4-Pakete, entpackt sie und leitet die IPv6-Pakete über die vorhandene IPv6-Infrastruktur weiter

Sicherheitsbedenken

Im Gegensatz zu Szenarien mit konfigurierten Tunneln kann man bei 6to4 keine Filter aufsetzen, um Pakete von unbekannten Rechnern zu filtern, da es sich bei genau diesen um entfernte 6to4 Gateways handeln könnte, die Daten/Antworten senden. 6to4 funktioniert dadurch, dass jeder entfernte Rechner in IPv4-Pakete eingepackte IPv6-Pakete senden kann, jedoch erlaubt dies auch böswilligen Zeitgenossen, Pakete mit ungültige oder absichtlich falsche Nutzlast zu senden. Falls nicht bereits geschehen sollten 6to4 Pakete mit den folgenden Merkmalen bereits beim Eintritt ins eigene Netzwerk von den Border-Routern identifiziert und verworfenden, z.B. durch passend konfigurierte Firewalls oder Paket-Filter:

- Pakete mit nicht spezifizierter IPv4 Quell/Zieladresse: 0.0.0.0/8
- Loopback Adresse als Quelle/Ziel: 127.0.0.0/8
- IPv4 Multicast Adresse als Quelle/Ziel: 224.0.0.0/4
- Broadcasts Adressen: 255.255.255.255
- Broadcast Adressen von Subnetzen: abhängig von den eigenen IPv4 Netzen

Die stf(4) Man-Page von NetBSD beschreibt die häufigsten Konfigurationsfehler wie sie vom KAME IPv6 Stack standardmässig abgefangen werden und gibt auch weitere Hinweise worauf man beim filtern von Paketen achten sollte. Zusammenfassend sollte man im Hinterkopf behalten dass 6to4 aufgrund seines Designs nicht absolut wasserdicht ist. Die in IPv6 eingebauten Sicherheitsfunktionen wie Authentifizierung und Verschlüsselung mittels IPsec können hier jedoch Abhilfe schaffen. Eine genauere Erklärung dieser Mechanismen sei hier auf einschlägige Literatur [Doraswamy], [Smith] verwiesen.

6to4 Setup - Zutaten

Um 6to4 selbst etwa von zu Hause über Dial-Up Internet oder im Büro mit Standleitung aufzusetzen, werden neben der bestehenden IPv4 Anbindung einige Daten dazu benötigt. Diese sollen im Folgenden kurz aufgelistet und erklärt werden. Wir beginnen mit der IPv4 Adresse des lokalen Netzwerk Interfaces. Unter Unix-artigen Betriebssystemen kann man diese üblicherweise mit den Befehlen "ifconfig -a" oder "netstat -i" ermitteln. Falls im IPv4-Netz Network Address Translation (NAT) - auch bekannt als IP Masquerading - eingesetzt wird, so muss 6to4 auf dem NAT-Gateway aufgesetzt werden, da auf jeden Fall eine weltweit gültige, offizielle IP-Nummer verwendet werden muss. Private Adressen aus den Netzen 10.0.0.0 oder 192.168.0.0 gehen hierzu leider nicht. Für die folgenden Beispiele wollen wir "62.224.57.114" als lokale IPv4 Adresse verwenden. Aus der IPv4-Adresse lässt sich die IPv6-Adresse bestimmen, wie weiter oben beschrieben - 62.224.57.114 ist hexadezimal gleich 3ee03972, als Subnetz-Adresse wählen wir 0001. Da wir im Beispiel keinen Auto-Konfigurierten Rechner sondern einen Gateway aufsetzen ist es erlaubt, die Host-Bits des Interfaces selbst zu wählen, anstatt sie von der MAC-Adresse abzuleiten. Wir verwenden "::1" als Host-Bits, so dass sich als vollständige IPv6 Adresse 2002:3ee0:3972:0001::1 ergibt.

Abhängig vom verwendeten IPv6-Stack wird als nächstes entweder die IPv4- oder die IPv6-Adresse des als Uplinks fungierenden 6to4 Gateways benötigt. Bei den KAME-basierenden BSD Stacks reicht die IPv6 Adresse, da diese ja die IPv4 Adresse des Gateways enthält. Verwendet man Linux, so sollte man zusätzlich die IPv4 Adresse bereithalten. Wir verwenden im Folgenden 2002:c25f:6cbf::1 (= 0xc25f6cbf = 194.95.108.191 = 6to4.ipv6.fh-regensburg.de).

Kernel backen

Um ein- und ausgehende 6to4 Pakete zu verarbeiten muss das Betriebssystem über diese bescheid wissen. Dazu ist ein Treiber für ein 6to4 Netzwerk-Interface nötigt, das die 6to4-Pakete zu verarbeiten weiss. Um BSD/KAME basierten Kernel IPv6 und 6to4 beizubringen sind die folgenden Zeilen in der Kernel-Konfigurationsdatei nötig:

```
options INET6              # IPv6
pseudo-device stf          # 6to4 IPv6 over IPv4 encapsulation
```

Man beachte dass der stf(4) Treiber, der für 6to4 zuständig ist, ggf. nicht per Default im Kernel enthalten ist.

Die Konfiguration unter Linux besteht aus einem "make config" bzw. "make menuconfig", und es ist sicherzustellen, dass die folgenden Optionen entsprechend beantwortet werden:

```
Networking options The IPv6 protocol:      yes
IPv6: enable EUI-64 token format:          yes
```

```
IPv6: disable provider based address:     yes
```

Nach der Konfiguration muss der BSD- oder Linux-Kernel und evtl. benötigte Kernel-Module übersetzt und installiert werden, und das System anschliessend neu gestartet werden, um die neuen Treiber zu benutzen. Für weitere Informationen zur Konfiguration, Compilierung und Installation sei auf die Dokumentation der verwendeten BSD/Linux Version verwiesen.

Wie sag ich's meinem Kinde

Die folgenden Befehle gelten für RedHat Linux 7.0 und NetBSD 1.5, da sie jedoch keine "magischen" Variablen des Startup-Systems des jeweiligen Betriebssystems verwendet sollten sie leicht übertragbar sein. Kurz zusammengefasst besteht die eigentlich Konfiguration aus den folgenden Schritten:

1. 6to4 Interface konfigurieren
2. Default-Route setzen
3. Ggf. Router Advertisement aufsetzen

Der erste Schritt beim Aufsetzen von 6to4 besteht darin, dem eigenen 6to4 Interface eine IPv6 Adresse zuzuweisen. Dies wird mit dem ifconfig(8) Befehl bewerkstelligt. Mit den oben genannten Beispieldaten ist dies für NetBSD:

```
ifconfig stf0 inet6 2002:3ee0:3972:1::1 prefixlen 16 alias  # eigene Adresse
```

Unter Linux sind zwei Befehle nötig, um die v4 und v6 Schichten des sit (Simple Internet Transition) Interfaces getrennt zu konfigurieren:

```
ifconfig sit0 tunnel ::194.95.108.191 up     (v4 Schicht, Adresse Gateway)
ifconfig sit1 add 2002:3ee0:3972:1::1/64     (v6 Schicht, eigene Adresse)
```

Nachdem das 6to4 Interface mit diesen Befehlen konfiguriert ist muss als nächstes dem System beigebracht werden, dass es sämtlichen IPv6 Traffic an den Relay Router sendet. Am einfachsten erreicht man dies über eine Default-Route, der Befehl dafür ist für NetBSD:

```
route add -inet6 default 2002:cdb2:5ac2::1 (v6 Adresse Relay Router)
```

und für Linux:

```
route -A inet6 add default gw ::194.95.108.191 (v4 Adresse Relay Router)
```

Nach diesen Befehlen ist man ans weltweite IPv6-Netz angeschlossen! Angenommen dass die Namensauflösung noch immer über IPv4 bewerkstilligt wird so kann man nun IPv6-Rechner wie z.B. www.kame.net oder www6.netbsd.org anpingen, für NetBSD lauten die Befehle

```
/sbin/ping6 www.kame.net
/sbin/ping6 www6.netbsd.org
```

Bei Linux erreicht man dasselbe mit folgendem Befehl:

```
/usr/ipv6/bin/ping -A inet6 www.kame.net
/usr/ipv6/bin/ping -A inet6 www6.netbsd.org
```

An dieser Stelle ist noch anzumerken dass das BSD/KAME stf-Interface die IPv4-Adresse des 6to4 Relay Routers aus der in der Routing-Tabelle enthaltenene Adresse ermittelt. Mit Hilfe dieses Features ist es sehr einfach, einen eigenen 6to4 Relay Router aufzusetzen, an den dann andere Rechner selbst 6to4 Pakete senden können, ohne sich vorher zu registrieren. Eine IPv6 Anbindung z.B. ans 6Bone ist dazu allerdings empfehlenswert, damit keine zu langen Tunnelwege entstehen.

Der letzte Schritt beim Konfigurieren von IPv6 ist das Router Advertisement. Dies ist nicht an 6to4 gebunden, und muss nur ausgeführt werden wenn man mehrere Rechner im lokalen Netz hat. Es ist zwar möglich, 6to4 auf jedem einzelnen Rechner zu konfigurieren (sofern er 6to4 IPv6 unterstützt!), jedoch führt dies zu sehr umständlichem Routing wenn man Daten von einen Rechner an den Nachbarrechner senden will. Die Pakete würden hier an den entfernten 6to4 Relay Router gehen, der dann wiederum die Pakete über 6to4 zurück an den Nachbarrechner senden würde. Stattdessen setzt man üblicherweise 6to4 nur auf einem Rechner als Router auf, spricht im lokalen Netz IPv6 und verwendet den Rechner mit 6to4 als Default-Gateway für IPv6.

Zum Konfigurieren von Router Advertisement muss zuerst dem Ethernet Interface eine IPv6 Adresse zugewiesen werden. Wir verwenden hier "0002" als Subnetz, und mit 12:34:56:78:9a:bc als MAC Adresse der Ethernet-Karte und unserem 6to4-Präfix ergibt sich die IPv6 Adresse zu 2002:3ee0:3972:2:1234:56ff:fe78:9abc. Diese weisen wir der Netzwerkkarte zu:

```
ifconfig ne0 inet6 alias 2002:3ee0:3972:2:1234:56ff:fe78:9abc prefixlen 64
```

für NetBSD, bzw. für Linux:

```
ifconfig eth0 add 2002:3ee0:3972:2:1234:56ff:fe78:9abc/64
```

Bei BSD ist in unserem Beispiel "ne0" die verwendete Netzwerkkarte. Da diese abhängig von der verwendeten Karte ist sollte hier mittels dmesg(8) und ifconfig(8) die wirklich verwendete Karte ermittelt werden, sofern nicht bereits bekannt. Bei Linux ist dies immer "eth0".

Der nächste Schritt ist die Konfiguration des Router Advertisement-Dämons. Dazu ist unter BSD die Datei /etc/rtadvd.conf zu überprüfen. Mit ihrer Hilfe können viele Parameter eingestellt werden, die nur aus Kommentaren bestehende Grundeinstellung reicht jedoch für die meisten Fälle. Mit ihnen werden die IPv6-Präfixe aller Netzwerkkarten an die jeweils angeschlossenen Rechner bekanntgegeben, und die Link-Local Adresse des jeweiligen Interfaces als möglicher Router für das jeweilige Subnetz angepriesen. Von dieser Ankündigung der Route hat das "Router Advertising" seinen Namen.

Unter Linux heisst die zuständige Datei /etc/radvd.conf, und in ihr muss u.a. der aktuelle Adresspräfix eingetragen sein. In unserem Beispiel:

```
interface eth0
{       AdvSendAdvert on;

        prefix 2002:3ee0:3972:2::/64
        {       AdvOnLink on;
                AdvRouterAddr on;
        };
};
```

Anschliessend ist sicherzustellen, dass Pakete vom 6to4 Interface ans Ethernet weitergereicht werden, und umgekehrt. Dies wird durch das Einschalten von Packet Forwarding bewirkt. Für NetBSD ist dazu "ip6mode=router" in der Datei /etc/rc.conf einzutragen. Als Resultat wird der "net.inet6.ip6.forwarding" sysctl auf "1" gesetzt wird, was auf allen BSD Systeme gleich funktioniert. Unter Linux ist sicherzustellen dass /proc/sys/net/ipv6/ip_forward auf "1" gesetzt ist:

```
BSD:    sysctl -w net.inet6.ip6.forwarding=1
Linux:  echo 1 >/proc/sys/net/ipv6/ip_forward
```

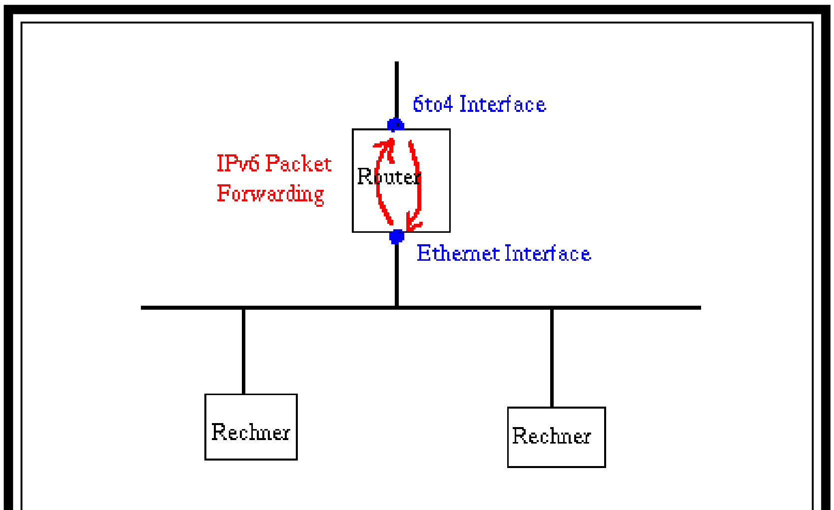

Abb.: Auf dem 6to4 Router muss Packet Forwarding für IPv6 eingeschaltet werden

Nachdem sichergestellt ist dass das Router Advertizement passend konfiguriert und Packet Forwarding für IPv6 angeschaltet ist kann der Dämon gestartet werden, der für das Router Advertisement zuständig ist. Unter NetBSD hört dieser auf den Namen "rtadvd", unter Linux heisst er "radvd". Er kann am Anfang manuell, später über den Start-Mechanismus des jeweiligen Betriebssystems gestartet werden. Sobald er gestartet ist kann man auf den am Netz angeschlossenen Rechnern, die für die Autokonfiguration via IPv6 vorbereitet sind beobachten, wie die Ethernet-Interfaces automatisch IPv6 Adressen zugewiesen bekommen, und unser 6to4-Router mit seiner Link-Local Adresse als Defaultrouter registriert ist.

Bekannte 6to4 Relay Router

Es gibt momentan nicht allzu viele öffentlich zugängliche 6to4 Relay Router. Von den wenigen vorhandenen wird man sich denjenigen aussuchen, der netztechnisch am nächsten ist. Eine Liste bekannter 6to4 Relay Router ist unter der URL http://www.kfu.com/~nsayer/6to4/ zu finden. Bei Tests hat sich gezeigt dass nur 6to4.kfu.com und 6to4.ipv6.microsoft.com verlässlich arbeiten. Cisco bietet einen eigenen 6to4 Relay Router an für dessen Benutzung man sich jedoch erst registrieren muss, nähere Informationen hierzu sind unter http://www.cisco.com/ipv6/ erhältlich. Desweiteren existiert zwei experimentelle 6to4 Relay Router in Deutschland, 6to4.ipv6.uni-leipzig.de und 6to4.ipv6.fh-regensburg.de. Letzterer läuft unter NetBSD 1.5 und wurde mit den oben beschriebenen Schritten konfiguriert. Die vollständige Konfiguration der Maschine ist under der Adresse http://www.feyrer.de/IPv6/netstart.local einzusehen.

Abgesehen von den festen Relay Routern wird in [RFC 3068] eine Anycast-Adresse 2002:c058:6301:: festgelegt, die jeweils auf den netzwerk-topologisch am nächsten gelegenen 6to4 Relay Router zeigt.

Dies erleichtert die Konfiguration von 6to4, da nicht mehr gesucht werden muß, welcher Router am nächsten oder am besten erreichbar ist, sondern diese Auswahl automatisch geschieht.
