

Automatische Netzwerk- Konfiguration von IPv6-Rechnern

Klaus Kusche, Juni 2013

Zum Titel ...

Eigentlich:

Automatische Netzwerk-Konfiguration
von *IPv6-Interfaces* (Netzwerk-Karten)

Denn:

- Konfiguration ist *pro Interface*,
nicht pro Rechner
- Interfaces sind voneinander *unabhängig*

Inhalt

- **Iststand IPv4**
- **Anforderungen**
- **IPv6 Grundlagen**
- **Autokonfiguration:
“stateless” (SLAAC)
“stateful” (DHCP)**
- **Probleme**

Was wird in IPv4 konfiguriert?

- IP-Adresse & Netzmaske
- Adresse des Routers / Gateways
- Liste der DNS-Server, Namen der eigenen Domain(s)
- MTU (nicht DHCP: Default oder händisch konfiguriert)

Optional:

- Hostname
- Lease Time (Gültigkeitsdauer der Adresse)
- Boot Image (bei "Boot from Net")
- Dienste: WINS, Time / NTP, WWW-Proxy, ...

Wie wird in IPv4 konfiguriert?

- Händisch
- Automatisch mittels DHCP
(Dynamic Host Configuration Protocol)
- Automatisch mittels APIPA
(Automatic Private IP Addressing)
= Zeroconf / Avahi / Bonjour (im Kern dasselbe)
- Ganz selten in Ausnahme-Fällen, nur für die Adresse:
Statisch remote mittels Reverse ARP

DHCP (1)

- Serverbasiert,

Server-Suche mittels Broadcast (nicht geroutet!)

==> Server (oder Proxy) in jedem Subnetz notwendig!

- Stateful

==> Server führt Buch über aktive Clients

==> Server-Abstürze oder -Wechsel sind ein Problem!

- Clients sollten sich abmelden,
sonst ev. Adressmangel!

==> Problem bei häufig wechselnden
oder “hart abgeschalteten” Clients

DHCP (2)

- (Aufwändige) Konfiguration des Servers nötig, außer im einfachsten Spezialfall (“DHCP zu Hause”):
 - Verwendung eines privaten Adressbereichs (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16)
 - + rein dynamische Adress-Vergabe, keine fixen / permanenten Client-Adressen
- ==> **Gut** für klassische, administrierte Rechnernetze (Firmennetz, Provider)
- ==> **Bedingt** geeignet für Durchschnitts-User (Heimnetz)
- ==> **Schlecht** z.B. für “Internet der Dinge”! (“dumme” Clients, volatile / spontane Netze)

Zeroconf etc. (1)

- “Hack” von Apple & Microsoft
=> Standardisierung 2004 erfolglos abgebrochen
- Ziel: Kleine ad-hoc-Netze, nur lokal (nicht Internet):
Mac findet Drucker, iTunes findet Musik-Devices,
Mini-PC-Netz: 2 PC's + Kreuzkabel (ohne Server), ...
- Peer-to-peer (“unter Gleichen”),
ohne speziellen Server
- De facto “stateless”

Zeroconf etc. (2)

Funktion / Ablauf:

- Wird aktiviert, wenn kein DHCP-Server erreichbar ist
- “Zufällige” Adresse in 169.254.x.x / 16 wählen
169.254.x.x ist per Definition

Subnetz-lokal = nicht routbar!

- Per ARP prüfen ob Adresse frei ist,
sonst nächster Versuch
- Weitere Info's, Suche nach Diensten:
Mittels Multicast (mDNS, DNS-SD)

Anforderungen IPv6 (1)

“Plug and Play Networking”

- Keine händische Eingabe:
 - IPv6 Adresse = 32 Hexziffern
=> zu fehleranfällig!
 - “Internet der Dinge”
 - => Oft keine Eingabe-Möglichkeit
 - => Oft keine permanente Speicher-Möglichkeit
 - Mobile Geräte
=> Häufig wechselnde Adressen!
- Wahlweise “klassisch” oder ohne Server & stateless

Anforderungen IPv6 (2)

Ist-Zustand IPv4 (wegen Adress-Knappheit & Sicherheit):

Meist private Adressen + NAT am Router
(10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16)

=> Adressvergabe "einfach":

Muss nur lokal eindeutig sein, *globale Gültigkeit irrelevant!*

Änderung IPv6:

*NAT ist explizit unerwünscht,
End-to-end-Adressierung gefordert!*

=> Jedes Gerät braucht eine global eindeutige Adresse

=> Provider teilt ganzen Adressbereich (Subnetz) zu!

IPv6 Adressen, allgemein

IPv6-Adresse = 128 Bits

Hinten:

64 Bit “Interface Identifier” (= Geräte-Adresse)

Meist lokal berechnet:

Aus MAC-Adresse (EUI-64) oder zufällig

Alternativ: Frei gewählt bzw. durch Konfig vorgegeben

Vorne:

64 Bit “Prefix” (= Netz-Adresse)

Vom Provider, global eindeutig

(oft: 32 Bit Provider, 16-24 Bit Kunde, 16-8 Bit intern)

“Interface Id” gemäß EUI-64

64 Bits,

berechnet aus der 48 Bit MAC-Adresse der Karte:

- Vordere 24 MAC-Bits
(= Hersteller-Kennung)
mit 7. Bit invertiert
- Fix **FFFE**
- Hintere 24 MAC-Bits
(= fortlaufende Karten-Nummer)

IPv6 Konfig: Was braucht man?

IPv4:

*Komplette Adresse (Netz- + Geräte-Adresse)
+ Netz-Maske*

IPv6:

Nur Prefix (Netz-Adresse) nötig!

- Keine Subnetz-Maske nötig (immer 64 + 64 Bits)
- Interface Id (Geräte-Adresse) ist optional:
 - Kann explizit konfiguriert werden
 - Wenn nicht: Wird automatisch berechnet

Spezielle IPv6-Adressen

- Prefix **fe80: . . .** : **“Link-lokale Adressen”**

==> Nur im eigenen Subnetz (bis zum Router) gültig

==> Nicht geroutet

==> Nur im Subnetz eindeutig, nicht weltweit

(entspricht in etwa 169.254.x.x von Zeroconf!)

- Prefix **ffxx: . . .** : **Multicast & Broadcast**

(xx ... Statusbits, z.B. für Gültigkeits-Bereich)

Zahlreiche im Standard festgelegt, hier wichtig:

ff02:2 ==> Alle Router im eigenen Subnetz

Adressen pro Interface

IPv4 im Normalfall:

Jedes Interface hat genau 1 Adresse

IPv6 im Normalfall (außer bei rein lokalem Netz):

Jedes Interface besitzt
mehrere unabhängige Adressen gleichzeitig

- Immer min. **1 Link-lokale** (parallel zu globalen!)
- Min. **1 globale**

Beispiele:

- Dyn. Adresswechsel: Alte & neue Adresse
- Mehrere Provider redundant: 1 Adresse pro Provider

Wie wird in IPv6 konfiguriert?

Händisch: Ev. bei Servern sinnvoll

(weil automat. / dyn. Adressen haben oft keine DNS-Namen,
dynDNS ist für IPv6 noch nicht etabliert!)

Automatisch:

- Neu, ähnlich Zeroconf: “stateless” = SLAAC (RFC 4862)

SLAAC = “Stateless Address Autoconfiguration”

Für kleine Netze, ad-hoc-Netze (privat, Gadgets, ...)

- Wie bisher: “stateful” = DHCPv6

Für große, administrierte Netze (Firmen)

- In der Praxis: Oft beides kombiniert (siehe “Probleme”)

Ablauf der autom. Konfiguration

1. Eigenständiges Aktivieren einer

link-lokalen Adresse

==> Knoten kann im eigenen Subnetz kommunizieren

2. Immer zuerst versuchen:

Router-Suche & SLAAC

3. Wenn erfolglos

oder Antwort "DHCP verwenden":

DHCP

Die Link-lokale Adresse

Vom Knoten selbst zusammengesetzt aus

- Link-lokalem Prefix **ffe8::0**
- Selbstberechneter "Interface Id"

Prüfung auf Eindeutigkeit mittels

"Neighbor Discovery Protocol" (RFC 2461)

Teil von ICMPv6

übernimmt die Funktion von ARP in IPv4

"frei" ==> Adresse aktivieren

"belegt" ==> andere Interface Id probieren oder aufgeben

SLAAC (1)

- NDP/ICMP “Router Solicitation” an **ff02::2** senden
- Router antwortet mit

“Router Advertisement” (RA)

RA enthält:

- SLAAC-Status
- Router-Adresse
- Wenn SLAAC aktiviert ist:
Globale Prefixes + deren Lebensdauer
- *MTU*

SLAAC (2)

=> Knoten entnimmt daraus die Router-Adresse

+

=> Knoten berechnet daraus seine

eigenen Adressen:

Globale Prefixes (aus RA)

+ eigene Interface Id

Vorsichtshalber für jede globale Adresse:

Prüfung ob eindeutig (mittels NDP)!

Mögliche SLAAC-Ergebnisse

- RA liefert Status “**SLAAC + kein DHCP**”
=> Router-Adresse + Prefixes verwenden, fertig!
- RA liefert Status “**SLAAC + stateless DHCP**” (siehe später)
=> Router-Adresse + Prefixes verwenden,
weitere Konfig von DHCP holen
- RA liefert Status “**DHCP, kein SLAAC**”
=> Nur Router-Adresse verwenden,
eigene Adresse + weitere Konfig von DHCP holen
- Kein RA innerhalb **Timeout**
=> DHCP versuchen!
- Eigene lokale oder globale Adresse **doppelt**
=> Aufgeben oder mit anderer Interface Id wiederholen

SLAAC ist “stateless”

Keine Buchführung über

- *Vergebene Adressen*
- *Aktive Clients*

==> Für den Client:

- Keine “Abmelde-Pflicht”

==> Für den Router (im Vergleich zu DHCP-Server):

- Reboot, Wechsel auf anderen Router, ... ist unkritisch
- Code einfacher, weniger Platzbedarf für Daten
- Weniger Konfiguration nötig (nur Prefixe, siehe später)

Weitere SLAAC-Vorteile

- Funktioniert in rein lokalen Netzen “peer-to-peer”
(ganz ohne Router / Server / Konfiguration)
- Clients sind robust gegen ausfallende Router
=> Suchen sich dynamisch einen anderen
- Standard sieht Lebensdauer
und dynamische Änderbarkeit
von Prefix & Interface Id vor
=> Mobile Devices!
=> Privacy (siehe “Probleme”)
- Fix im IPv6-Code des Betriebssystems
=> Kein eigener User-Mode Client- / Server-Code nötig

DHCPv6

- Ähnlich zu DHCPv4
- Parallel mit DHCPv4 betreibbar:
Andere Ports
- Erreichbar über fixe, lokale Multicast-Adressen:
ff02::1:2 und **ff05::1:3**
- DHCP-Relay fix standardisiert
- Liefert weitere optionale Dienst-Informationen,
z.B. SIP (Telefon-Server)

Problem 1: SLAAC ist zu wenig! (1)

SLAAC liefert nur *Adresse & Router*:

- Keine DNS-Info
(Server + Domainnamen)

Unbedingt nötig!

(außer wenn fix händisch konfiguriert)

- Keine optionale DHCP-Info
(Hostname, NTP, WINS, ...)

Problem 1: SLAAC ist zu wenig! (2)

Lösungen:

- SLAAC + “stateless DHCPv6” (RFC 3736):

- SLAAC liefert Adresse + Router
- DHCPv6 liefert alles andere

==> DHCP braucht keine Client-Buchführung mehr!

- Protokoll-Erweiterung von SLAAC bzw. RA:
RA liefert auch DNS-Info (RFC 6106, RDNSS + DNSSL)

Erst seit 2010, bisher kaum implementiert (nur Linux?)

Kein DHCP mehr nötig!

- Microsoft, inzwischen wieder verworfen:
DNS-Server via fixer, site-lokaler Anycast-Adresse erreichbar

Problem 2: DHCPv6 ist zu wenig!

Keine Router- / Gateway-Adresse

im DHCPv6-Protokoll vorgesehen!

==> Auch bei Adressvergabe via DHCP:

SLAAC / RA für Router-Info trotzdem nötig!

==> Protokoll-Erweiterung von DHCP
in (ferner) Diskussion

Problem 3: Privacy (1)

Beide Teile von SLAAC-Adressen sind für sich allein

- eindeutig
- Langzeit-stabil
- leicht Kunden bzw. Geräten zuzuordnen

weil

- Interface-Id ist aus MAC berechnet (EUI-64)
=> bleibt weltweit gleich, einem Gerät zuzuordnen
- Netz-Prefix wird vom Provider zugeteilt
=> Jeder Netzverkehr einem Kunden zuzuordnen

Problem 3: Privacy (2)

=> Werbeindustrie freut sich

=> Sicherheitsdienste & Rechteinhaber freuen sich

Lösungen:

- DHCP verwenden:
Explizit wechselnde Interface Id konfigurieren
- Provider rotiert Prefixes (z.B. täglich)
- Interface Id's nach RFC 4941 "Privacy Extensions"

Problem 3: Privacy (3)

Privacy Extensions:

- Eine fixe, gleichbleibende Interface Id für einlangende Verbindungen
- + mehrere “zufällige”, wechselnde (Stunden / Tage) für ausgehende Verbindungen
- Mittels MD5 Hash aus MAC + voriger Id (+ Zufall)
 - ==> Nicht vorhersehbar / zuzuordnen
 - ==> Minimierung der Kollisions-Wahrscheinlichkeit
- Bei jeder neuen Interface Id:
Prefix davor, Kollisions-Prüfung machen

Problem 4: Router-Konfig

Router müssen Netz-Prefixes wissen

=> Router brauchen (minimale) Konfiguration,
sind nicht "Plug and Play"

Lösung:

- Händisch
- Dynamische Router-Protokolle (ähnlich IPv4)
- DHCPv6 "Prefix Delegation" verteilt Prefix-Liste
z.B. vom Provider an die Kunden-Router

Problem 5: DHCPv6 “DUID”

“DUID” = eindeutige Geräte-Id
(bei Systeminstallation berechnet)

identifiziert den Client bei DHCP-Requests
(statt der MAC-Adresse bei DHCP v4)

Vorteile:

- “Überlebt” Tausch der Netzwerk-Karte
- Immun gegen mehrere Pfade (z.B. LAN / WLAN)

Nachteile:

- Ändert sich bei System-Neuinstallation
- Bei Multiboot-Systemen: In jeder Installation anders!

“The end”

Fragen?