Engineer

IPv6

versionen av internetprotokollet med adresser på 128 bitar

Vad det är

IPv6, utskrivet Internet Protocol version 6, är den version av internetprotokollet som togs fram som efterträdare till IPv4, den version som RFC 791 beskriver. Den gällande specifikationen är RFC 8200 från juli 2017, en internetstandard (STD 86) som ersatte RFC 2460. Den förändring som alla lägger märke till är adressens storlek: 128 bitar i stället för IPv4:s 32. Hela IPv4 rymmer 4 294 967 296 adresser; ett enda IPv6-subnät av den vanliga storleken, ett /64, rymmer det talet i kvadrat.

Även resten av paketformatet har gjorts om. Enligt RFC 8200 har några av IPv4:s huvudfält tagits bort eller gjorts valfria för att göra den vanliga behandlingen billigare, valfri information ligger i separata tilläggshuvuden efter själva huvudet, och fältet Time to Live heter nu Hop Limit och räknas ned med ett av varje nod som vidarebefordrar paket. Det finns ingen kontrollsumma i huvudet. Routrar delar aldrig upp ett paket: I IPv6 fragmenteras det bara hos avsändaren, varje länk måste klara paket på minst 1280 oktetter, och avsändare uppmanas starkt att använda Path MTU Discovery för att ta reda på hur stora paket en väg klarar.

En adress skrivs som åtta grupper om 16 bitar i hexadecimal form, åtskilda av kolon. RFC 4291 tillåter att man stryker inledande nollor i en grupp och ersätter en sammanhängande följd av nollgrupper med ett dubbelkolon, som bara får förekomma en gång, så att 2001:db8:0:0:8:800:200c:417a kan skrivas 2001:db8::8:800:200c:417a. Eftersom det ger många sätt att skriva samma adress väljer RFC 5952 från augusti 2010 ett av dem: gemener, inga inledande nollor och dubbelkolonet på den längsta följden av nollgrupper (den första om två är lika långa), aldrig på en ensam grupp. Bortsett från särskilda adresser som loopback-adresserna och de länklokala kommer alla exempeladresser och -prefix i den här artikeln från ett dokumentationsområde: 2001:db8::/32, som RFC 3849 har avsatt, eller för IPv4 de block som RFC 5737 reserverar.

RFC 4291 ändrar också vad en adress sitter på. Adresser tilldelas gränssnitt, inte maskiner, och ett gränssnitt kan ha flera samtidigt. Varje gränssnitt måste ha en länklokal adress under fe80::/10, som bara gäller på dess egen länk. Loopback-adressen är ::1. Det finns ingen broadcast; multicast har tagit över den uppgiften, och ff02::1 når alla noder på länken. Unika lokala adresser under fc00::/7, definierade i RFC 4193, är avsedda för lokal kommunikation, vanligtvis inom en organisations nät. Globala unicast-adresser används för kommunikation utanför det lokala nätet. För de flesta adresser är de sista 64 bitarna gränssnittets identifierare (interface identifier), och därför är ett vanligt nätsegment ett /64.

Värdar numrerar oftast sig själva. Med tillståndslös autokonfiguration av adresser (stateless address autoconfiguration, RFC 4862) annonserar routrar en länks prefix i periodiska Router Advertisements, varje värddator kombinerar ett prefix med identifieraren för sitt eget gränssnitt, och innan den tar resultatet i bruk kör den Duplicate Address Detection och skickar Neighbor Solicitation-meddelanden för att kontrollera att ingen annan nod har adressen. DHCPv6 är alternativet när man behöver striktare kontroll över exakt vilka adresser som delas ut, och de två kan köras sida vid sida.

Använd begreppet i ett beslut

Använd ”Vad är IPv6?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Wi-Fi, brandväggar och fjärråtkomst uppsatta så att personalen kan arbeta varifrån som helst och bara rätt personer kommer in.

Se våra tjänster Hör av er →

Granska antagandena bakom ordet

Använd förklaringen och det länkade arbetet för att formulera frågor till projektet. Vi kan undersöka antagandena tillsammans innan ett verktyg väljs. Förvaltade 30 VPS-instanser med Ubuntu Linux — 30 förvaltade Linux-servrar, med en plan för återställning

Förvaltade 30 VPS-instanser med Ubuntu Linux Hör av er →

Varför det spelar roll

För många team är IPv6 inte ett projekt som de bestämmer sig för att starta. Poängen med RFC 4862 är att ingen behöver knappa in en adress på en värddator: Ställ en server på en länk där en router annonserar ett prefix, så kan den själv få en global IPv6-adress. Varje tillåtelselista, brandväggsregel, loggparser eller hälsokontroll som är skriven med bara IPv4 i tankarna har då en blind fläck som är exakt lika stor som den nya adressrymden.

Den blinda fläcken är lätt att missa, eftersom de två versionerna är separata protokoll som inte delar något på tråden. Ett paketfilter matchar deras huvuden var för sig, en hastighetsbegränsning som räknar per IPv4-adress ser aldrig en

IPv6-klient, och ett test som bara ansluter till 127.0.0.1 säger ingenting om ::1. En värddator som är låst på den ena versionen och öppen på den andra har i praktiken ingen brandvägg.

IPv6 lutar sig också mer mot sina kontrollmeddelanden än IPv4 gör. Neighbor Discovery, routerannonseringar och Path MTU Discovery går alla över ICMPv6, så en vana som man tar med sig från IPv4, att blockera ICMP i nätets utkant, förstör IPv6 på sätt som kanske först märks när en förbindelse börjar flytta data. Och eftersom en adress kan stavas på många giltiga sätt blir kod som jämför adresser som text oense med sig själv.

Så ser det ut i praktiken

Pythons standardmodul `ipaddress` tolkar varje giltig stavning och skriver ut en kort form där inledande nollor är borttagna och den längsta följden av nollgrupper är sammanslagen, så det är ett snabbt sätt att planera ett prefix och att kontrollera att två stavningar är samma adress. Sessionen nedan är den här artikelns illustration, körd på Python 3.15 med adresser från dokumentationsprefixet:

```
>>> import ipaddress
>>> site = ipaddress.ip_network("2001:db8:42::/48")
>>> site.num_addresses == 2**80
True
>>> lans = site.subnets(new_prefix=64)
>>> next(lans), next(lans)
(IPv6Network('2001:db8:42::/64'), IPv6Network('2001:db8:42:1::/64'))
>>> len(list(site.subnets(new_prefix=64)))
65536
>>> a = ipaddress.ip_address("2001:0DB8:0000:0000:0001:0000:0000:0001")
>>> a
IPv6Address('2001:db8::1:0:0:1')
>>> a.exploded
'2001:0db8:0000:0000:0001:0000:0000:0001'
>>> a == ipaddress.ip_address("2001:db8:0:0:1::1")
True
>>> ipaddress.ip_address("fe80::1%eth0").scope_id
'eth0'
```

Ett /48 lämnar 80 bitar lediga, och uppdelat i /64-segment ger det 65 536 stycken, numrerade i den fjärde gruppen. Adressen som skrevs med versaler och alla nollor utskrivna kommer tillbaka i den korta formen, 2001:db8::1:0:0:1, där dubbelkolonet täcker den första av två lika långa följder av nollor, och den är lika

med 2001:db8:0:0:1::1, en annan men giltig stavning av samma 128 bitar. Den sista raden visar en länklokal adress med sin zon, det `%eth0` som säger vilken länk den hör till.

På en Linux-värd med nftables är illustrationens andra del en regeluppsättning som släpper in SSH från ett kontor över båda versionerna. Manualsidan nft(8) beskriver `inet` som en hybridfamilj för tabeller som rymmer både IPv4 och IPv6, medan `ip saddr` läser IPv4-huvudet och `ip6 saddr` IPv6-huvudet:

```
add table inet filter
add chain inet filter input { type filter hook input priority filter; }
add rule inet filter input ip saddr 198.51.100.0/24 tcp dport 22 accept
add rule inet filter input ip6 saddr 2001:db8:42::/48 tcp dport 22 accept
add rule inet filter input tcp dport 22 drop
```

Utan den fjärde raden träffar en administratör som ansluter från kontorets IPv6-prefix ingen av accept-reglerna och blir avvisad, och det felet märks åtminstone direkt. Det tysta felet är det omvända: En regeluppsättning som bara är byggd i familjen `ip`, på en värddator som också har en IPv6-adress, tittar aldrig på IPv6-trafiken alls.

Portföljposten Provisionerade en andra server från kod från september 2026 tar upp samma exponering från provisioneringssidan. Den första maskinen hade skapats för hand och fått sin brandvägg i efterhand, vilket lämnade ett fönster där en ny värd med en standardpolicy för lösenord var nåbar från internet. Den andra skapas först efter sin brandvägg: Brandväggens play körs före serverns play, och båda leverantörernas brandväggar skrivs direkt mot deras REST-API:er. Sidan nämner inte IPv6, så reglerna ovan är den här artikelns illustration och inte den värdens regler. Det som den här artikeln lägger till är att brandväggen måste täcka varje adress som värddatorn kommer att ha, IPv6 inräknat, innan värddatorn ens finns.

Var det går fel

En brandvägg som bara känner till IPv4. Regler skrivna i familjen `ip`, eller tillåtelselistor med bara IPv4-prefix, lämnar alla värdens IPv6-adresser åt det som kommer efteråt. Skriv regeluppsättningen i familjen `inet`, eller håll `ip` - och `ip6` -tabeller i takt, sätt en `ip6 saddr` -regel bredvid varje `ip saddr` -regel, och testa varje tjänst genom att ansluta till dess IPv6-adress lika väl som till dess IPv4-adress.

Att blockera all ICMPv6. RFC 4890, som handlar om att filtrera de här meddelandena i brandväggar, säger det rakt ut: ”ICMPv6 är avgörande för att IPv6 ska fungera”. En router som inte kan vidarebefordra ett paket för att det är för stort för nästa länk kastar det och skickar tillbaka ett Packet Too Big-meddelande, och RFC 8201 påpekar att om filtrering stoppar de meddelandena får avsändaren aldrig veta vägens MTU. Resultatet är vad RFC:n kallar en black hole-förbindelse: TCP-handskakningen lyckas, och sedan hänger överföringen. RFC 4890 räknar också upp Neighbor Solicitation och Advertisement och Router Solicitation och Advertisement som meddelanden som en nod inte får blockera om den ska kunna delta i ett IPv6-nätverk. Acceptera de typer som nft kallar `packet-too-big`, `nd-neighbor-solicit`, `nd-neighbor-advert`, `nd-router-solicit` och `nd-router-advert` före en bredare drop-regel.

Att jämföra adresser som text. RFC 5952 ger exemplet med ett kalkylblad där en sökning efter 2001:db8:0:0:1::1 missar en post som skrivits 2001:db8::1:0:0:1, vilket spelar roll när listan genomsöks innan en ny adress delas ut, för att kontrollera att den inte redan används. Tolka adresserna innan du jämför dem, och lagra och skriv ut en kanonisk form, till exempel den korta form som Pythons `ipaddress` returnerar från `str()`.

En adress och en port ihopskrivna. 2001:db8::1:80 kan vara en adress eller en adress följd av port 80; RFC 5952 kallar den formen tvetydig och rekommenderar hakparenteser, [2001:db8::1]:80, som i URL:er. Skriv alltid parenteserna, och dela i kod upp värd och port vid den avslutande parentesen i stället för vid det sista kolonet.

Ett länkprefix som inte är ett /64. Autokonfiguration kräver att prefixet och gränssnittets identifierare tillsammans blir 128 bitar, och med 64 bitar långa identifierare bortser en värddator enligt RFC 4862 från ett annonserat prefix av någon annan längd, så värdar på en länk vars Router Advertisements bär ett /72 eller ett /56 bildar ingen adress av det prefixet. Planera varje länk som ett /64, och dela upp större tilldelningar i /64-block, så som sessionen ovan gör.

Länklokala adresser utan sin zon. Varje gränssnitt har en adress i fe80::/10, och RFC 4291 begränsar de adresserna till en enda länk, så på en värddator med flera gränssnitt säger adressen ensam inte vilken länk som avses. Pythons dokumentation påpekar att två adresser som bara skiljer sig i sin zon inte är lika. Håll ihop zonen med adressen (fe80::1%eth0), eller ge tjänster en global eller unik lokal adress och låt de länklokala stanna på länken.

Läs mer

- RFC 8200: Internet Protocol, Version 6 (IPv6) Specification — den gällande IPv6-standarden, med huvudformatet, tilläggshuvudena och reglerna för fragmentering och paketstorlek.
- RFC 4291: IP Version 6 Addressing Architecture — adresstyperna, textformerna, länklokala adresser och multicastadresser samt 64 bitar långa identifierare för gränssnitt.
- RFC 5952: A Recommendation for IPv6 Address Text Representation — det kanoniska sättet att skriva en adress och de problem som många stavningar ställer till med.
- RFC 4862: IPv6 Stateless Address Autoconfiguration — hur värdar bildar sina egna adresser av annonserade prefix och kontrollerar dem mot dubbletter.
- RFC 4890: Recommendations for Filtering ICMPv6 Messages in Firewalls — vilka ICMPv6-meddelanden en brandvägg måste släppa igenom, och varför.
- ipaddress: IPv4/IPv6 manipulation library — Pythons referens för att tolka, jämföra och dela upp IPv6-adresser och -nätverk.

Ordlista

Välj utifrån era faktiska förutsättningar

Ta med driftsättningsvägen, driftsriskerna och dem som har jour. Vi kan avgränsa en förändring med kontroller, en väg tillbaka och tydligt ansvar.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är IPv6?” gäller i er organisation. Ett konkret exempel från ert nuvarande system räcker för att börja samtalet.

Se våra tjänster

welcome@engineer.company +45 50 10 61 84 Köpenhamn, Danmark

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/glossary/ipv6/