Engineer

IPv6

versjonen av internettprotokollen med adresser på 128 bit

Hva det er

IPv6, skrevet helt ut Internet Protocol version 6, er den versjonen av internettprotokollen som ble utformet som etterfølger for IPv4, versjonen som RFC 791 beskriver. Den gjeldende spesifikasjonen er RFC 8200 fra juli 2017, en internettstandard (STD 86) som erstattet RFC 2460. Endringen alle legger merke til, er størrelsen på en adresse: 128 bit i stedet for IPv4s 32. Hele IPv4 rommer 4 294 967 296 adresser; ett enkelt IPv6-subnett av den vanlige størrelsen, en /64, rommer det tallet i andre potens.

Resten av pakkeformatet er også endret. Ifølge RFC 8200 er noen av IPv4s headerfelt fjernet eller gjort valgfrie for å gjøre den vanlige behandlingen billigere, valgfri informasjon ligger i egne utvidelsesheadere etter selve headeren, og feltet Time to Live heter nå Hop Limit og telles ned med én av hver node som videresender pakker. Det finnes ingen kontrollsum i headeren. Rutere deler aldri opp en pakke: I IPv6 fragmenteres det bare hos avsenderen, hver lenke må kunne bære pakker på minst 1280 oktetter, og avsendere oppfordres sterkt til å bruke Path MTU Discovery for å finne ut hvor store pakker en sti kan bære.

En adresse skrives som åtte grupper på 16 bit i heksadesimal, skilt med kolon. RFC 4291 tillater at man sløyfer innledende nuller i en gruppe og erstatter én sammenhengende rekke med nullgrupper med et dobbelt kolon, som bare kan forekomme én gang, slik at 2001:db8:0:0:8:800:200c:417a kan skrives 2001:db8::8:800:200c:417a. Siden det gir mange måter å skrive den samme adressen på, velger RFC 5952 fra august 2010 én av dem: små bokstaver, ingen innledende nuller og det doble kolonet på den lengste rekken med nullgrupper (den første hvis to er like lange), aldri på en enkelt gruppe. Bortsett fra spesielle adresser som loopback-adressene og de lenkelokale kommer alle

eksempeladressene og -prefiksene i denne artikkelen fra et dokumentasjonsområde: 2001:db8::/32, som RFC 3849 har satt av, eller for IPv4 blokkene som RFC 5737 reserverer.

RFC 4291 endrer også hva en adresse sitter på. Adresser tildeles grensesnitt, ikke maskiner, og ett grensesnitt kan ha flere samtidig. Hvert grensesnitt må ha en lenkelokal adresse under fe80::/10, som bare gjelder på dets egen lenke. Loopback-adressen er ::1. Det finnes ingen kringkasting (broadcast); multicast har overtatt den jobben, og ff02::1 når alle noder på lenken. Unike lokale adresser under fc00::/7, definert i RFC 4193, er ment for lokal kommunikasjon, vanligvis innenfor én organisasjons nett. Globale unicast-adresser brukes til kommunikasjon utenfor det lokale nettet. For de fleste adresser er de siste 64 bitene grensesnittets identifikator (interface identifier), og derfor er et vanlig nettverkssegment en /64.

Verter nummererer som regel seg selv. Med tilstandsløs autokonfigurasjon av adresser (stateless address autoconfiguration, RFC 4862) kunngjør rutere prefiksene på en lenke i periodiske Router Advertisements, hver vert kombinerer et prefiks med identifikatoren til sitt eget grensesnitt, og før den tar resultatet i bruk, kjører den Duplicate Address Detection og sender Neighbor Solicitation-meldinger for å sjekke at ingen annen node har adressen. DHCPv6 er alternativet når man trenger strammere kontroll over nøyaktig hvilke adresser som deles ut, og de to kan kjøre side om side.

Bruk begrepet i en beslutning

Bruk «Hva er IPv6?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Wi-Fi, brannmurer og fjerntilgang satt opp slik at de ansatte kan jobbe hvor som helst, og bare de rette personene slipper inn.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Administrerte 30 Ubuntu Linux-VPS-instanser — 30 Linux-servere administrert, med gjenoppretting planlagt på forhånd

Administrerte 30 Ubuntu Linux-VPS-instanser Ta kontakt →

Hvorfor det er viktig

For mange team er IPv6 ikke et prosjekt de bestemmer seg for å starte. Poenget med RFC 4862 er at ingen trenger å taste inn en adresse på en vert: Sett en server på en lenke der en ruter kunngjør et prefiks, og den kan selv få en global

IPv6-adresse. Enhver tillatelsesliste, brannmurregel, loggparser eller helsesjekk som er skrevet med bare IPv4 i tankene, har da en blindsone som er nøyaktig like stor som det nye adresserommet.

Den blindsonen er lett å overse, fordi de to versjonene er separate protokoller som ikke deler noe på ledningen. Et pakkefilter matcher headerne deres hver for seg, en hastighetsbegrensning som teller per IPv4-adresse, ser aldri en IPv6-klient, og en test som bare kobler til 127.0.0.1, sier ingenting om ::1. En vert som er låst ned på den ene versjonen og åpen på den andre, har i praksis ingen brannmur.

IPv6 lener seg også mer på kontrollmeldingene sine enn IPv4 gjør. Neighbor Discovery, ruterkunngjøringer og Path MTU Discovery går alle over ICMPv6, så en vane man tar med seg fra IPv4, nemlig å droppe ICMP i kanten av nettet, ødelegger IPv6 på måter som kanskje først viser seg når en forbindelse begynner å flytte data. Og fordi én adresse kan staves på mange gyldige måter, blir kode som sammenligner adresser som tekst, uenig med seg selv.

Slik ser det ut i praksis

Pythons standardmodul `ipaddress` tolker enhver gyldig stavemåte og skriver ut en kort form der innledende nuller er fjernet og den lengste rekken med nullgrupper er slått sammen, så det er en rask måte å planlegge et prefiks på og å sjekke at to stavemåter er den samme adressen. Økten nedenfor er denne artikkelens illustrasjon, kjørt på Python 3.15 med adresser fra dokumentasjonsprefikset:

```
>>> 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'
```

En /48 lar 80 bit stå ledige, og delt opp i /64-segmenter gir den 65 536 av dem, nummerert i den fjerde gruppen. Adressen som er tastet med store bokstaver og alle nuller skrevet ut, kommer tilbake i den korte formen, 2001:db8::1:0:0:1, der det doble kolonet dekker den første av to like lange rekker med nuller, og den er lik 2001:db8:0:0:1::1, en annen, men gyldig stavemåte for de samme 128 bitene. Den siste linjen viser en lenkelokal adresse med sin sone, den `%eth0` som sier hvilken lenke den hører til.

På en Linux-vert med nftables er den andre delen av illustrasjonen et regelsett som slipper inn SSH fra et kontor over begge versjonene. Manualsiden nft(8) beskriver `inet` som en hybrid familie for tabeller som rommer både IPv4 og IPv6, mens `ip saddr` leser IPv4-headeren og `ip6 saddr` IPv6-headeren:

```
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
```

Uten den fjerde linjen treffer en administrator som kobler til fra kontorets IPv6-prefiks, ingen av accept-reglene og blir avvist, og den feilen melder seg i det minste med en gang. Den stille feilen er den omvendte: Et regelsett som bare er bygd i familien `ip`, på en vert som også har en IPv6-adresse, ser aldri på IPv6-trafikken i det hele tatt.

Porteføljeoppføringen Klargjorde en server nummer to fra kode fra september 2026 tar for seg den samme eksponeringen fra klargjøringssiden. Den første maskinen var opprettet for hånd og hadde fått brannmuren sin etterpå, noe som etterlot et vindu der en fersk vert med en standard passordpolicy kunne nås fra internett. Den andre opprettes først etter brannmuren sin: Brannmur-playet kjører før server-playet, og begge leverandørenes brannmurer er skrevet direkte mot REST-API-ene deres. Siden nevner ikke IPv6, så reglene ovenfor er denne artikkelens illustrasjon og ikke den vertens regler. Det denne artikkelen legger til, er at brannmuren må dekke alle adressene verten kommer til å ha, IPv6 medregnet, før verten i det hele tatt finnes.

Hvor det går galt

En brannmur som bare kjenner IPv4. Regler skrevet i familien `ip`, eller tillatelseslister med bare IPv4-prefikser, overlater alle vertens IPv6-adresser til det som kommer etterpå. Skriv regelsettet i familien `inet`, eller hold `ip`- og `ip6`-tabeller i takt, sett en `ip6 saddr`-regel ved siden av hver `ip saddr`-regel, og test hver tjeneste ved å koble til IPv6-adressen dens i tillegg til IPv4-adressen.

Å droppe all ICMPv6. RFC 4890, som handler om å filtrere disse meldingene i brannmurer, sier det rett ut: «ICMPv6 er avgjørende for at IPv6 skal fungere». En ruter som ikke kan videresende en pakke fordi den er for stor for neste lenke, forkaster den og sender en Packet Too Big-melding tilbake, og RFC 8201 påpeker at hvis filtrering stopper de meldingene, får avsenderen aldri vite stiens MTU. Resultatet er det RFC-en kaller en black hole-forbindelse: TCP-håndtrykket lykkes, og så henger overføringen. RFC 4890 nevner også Neighbor Solicitation og Advertisement og Router Solicitation og Advertisement som meldinger en node ikke må droppe hvis den skal kunne delta i et IPv6-nettverk. Godta typene som nft kaller `packet-too-big`, `nd-neighbor-solicit`, `nd-neighbor-advert`, `nd-router-solicit` og `nd-router-advert`, før en bredere drop-regel.

Å sammenligne adresser som tekst. RFC 5952 gir et eksempel med et regneark der et søk etter 2001:db8:0:0:1::1 ikke finner en oppføring skrevet 2001:db8::1:0:0:1, og det betyr noe når listen søkes gjennom før en ny adresse deles ut, for å sjekke at den ikke allerede er i bruk. Tolk adressene før du sammenligner dem, og lagre og skriv ut én kanonisk form, for eksempel den korte formen som Pythons `ipaddress` returnerer fra `str()`.

En adresse og en port skrevet i ett. 2001:db8::1:80 kan være en adresse eller en adresse etterfulgt av port 80; RFC 5952 kaller den formen tvetydig og anbefaler hakeparenteser, [2001:db8::1]:80, som i URL-er. Skriv alltid parentesene, og del vert og port i kode ved den avsluttende parentesen i stedet for ved det siste kolonet.

Et lenkeprefiks som ikke er en /64. Autokonfigurasjon krever at prefikset og grensesnittets identifikator til sammen utgjør 128 bit, og med identifikatorer på 64 bit ignorerer en vert ifølge RFC 4862 et kunngjort prefiks av enhver annen lengde, så verter på en lenke der Router Advertisements bærer en /72 eller en /56, danner ingen adresse fra det prefikset. Planlegg hver lenke som en /64, og del større tildelinger opp i /64-blokker, slik økten ovenfor gjør.

Lenkelokale adresser uten sonen sin. Hvert grensesnitt har en adresse i fe80::/10, og RFC 4291 begrenser de adressene til én enkelt lenke, så på en vert med flere grensesnitt sier ikke adressen alene hvilken lenke som er ment. Pythons dokumentasjon påpeker at to adresser som bare skiller seg i sonen, ikke er like. Hold sonen sammen med adressen (fe80::1%eth0), eller gi tjenester en global eller unik lokal adresse, og la de lenkelokale bli på lenken.

Les mer

- RFC 8200: Internet Protocol, Version 6 (IPv6) Specification — den gjeldende IPv6-standarden, med headerformatet, utvidelsesheaderne og reglene for fragmentering og pakkestørrelse.
- RFC 4291: IP Version 6 Addressing Architecture — adressetypene, tekstformene, lenkelokale adresser og multicastadresser og identifikatorer på 64 bit for grensesnitt.
- RFC 5952: A Recommendation for IPv6 Address Text Representation — den kanoniske måten å skrive en adresse på, og problemene som mange stavemåter skaper.
- RFC 4862: IPv6 Stateless Address Autoconfiguration — hvordan verter danner sine egne adresser ut fra kunngjorte prefikser og sjekker dem for duplikater.
- RFC 4890: Recommendations for Filtering ICMPv6 Messages in Firewalls — hvilke ICMPv6-meldinger en brannmur må slippe gjennom, og hvorfor.
- ipaddress: IPv4/IPv6 manipulation library — Pythons referanse for å tolke, sammenligne og dele opp IPv6-adresser og -nettverk.

Ordliste

Velg ut fra deres faktiske rammer

Ta med utrullingsløpet, driftsrisikoene og dem som har vakt. Vi kan avgrense en endring med kontroller, en vei tilbake og tydelig ansvar.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er IPv6?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

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

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

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

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/ipv6/