Engineer

Nätverksprotokoll

de överenskomna regler två program följer för att tala

Vad det är

Ett nätverksprotokoll är en nedskriven överenskommelse om ett samtal mellan program: hur varje meddelande ser ut, vad det betyder, vilken sida som talar först, vad som får följa på vad och vad den ena sidan gör när den andra säger något oväntat. Två program som har skrivits av olika personer i olika språk kan arbeta ihop eftersom vart och ett byggdes mot samma dokument och inte mot det andra programmet. I vardagligt tal säger man oftast bara protokoll.

NIST:s ordlista samlar flera definitioner ur myndighetens publikationer, och de delar en kärna: regler för meddelandenas format och för procedurerna kring dem, som två eller fler parter följer för att nå ett gemensamt mål. En av definitionerna nämner den halva som är lätt att glömma: reglerna beskriver meddelandenas ordning lika väl som deras struktur. Ett meddelande i rätt format vid fel tillfälle bryter samtalet lika säkert som ett felformaterat.

Specifikationer som publiceras som RFC uttrycker sina regler med nyckelord i versaler. MUST är ett absolut krav. SHOULD tillåter en avvikelse bara när alla konsekvenser av den är förstådda och vägda. MAY markerar något som verkligen är valfritt. Sedan 2017 har orden den betydelsen enbart i versaler, så ett ”must” med gemener i samma dokument är vanlig engelska. Avståndet mellan SHOULD och MUST är avståndet mellan ett program som är ovanligt och ett program som är fel.

Protokoll staplas på varandra. Ett applikationsprotokoll, den del som känner till sidor, menyer eller e-post, åker oftast ovanpå ett transportprotokoll vars enda uppgift är att flytta byte mellan två maskiner. TCP, den transport som de flesta av dem använder, erbjuder det som specifikationen kallar ”a reliable, in-order, byte-stream service”: byten kommer fram fullständiga och i den ordning de skickades. Däremot lovar TCP inget samband mellan de bitar som den ena sidan skriver och

de bitar som den andra sidan läser. En ström består bara av byte, så varje protokoll ovanpå måste själv markera var dess meddelanden slutar, med ett radslut, en längd som skickas först, att anslutningen stängs eller någon annan överenskommen markör. UDP gör den motsatta uppgörelsen. Det bär separata datagram med ett minimum av mekanik, och varken leverans eller skydd mot dubbletter garanteras. Båda transporterna skiljer tjänsterna på en maskin åt med en port, och de numrerar sina portar var för sig.

Resten av ett protokolls karaktär kommer från några val som varje design gör, med avsikt eller inte:

- **Vem som talar först.** Vissa servrar hälsar på klienten, andra väntar tyst på en begäran, och när båda sidor väntar händer ingenting förrän en timer löper ut.
- **Vad servern minns.** I ett tillståndslöst protokoll bär varje begäran allt som behövs för att besvara den. I ett tillståndsbärande beror ett meddelandes betydelse på vad som kom före det på samma anslutning.
- **Hur det växer.** Versionsnummer, koder för tillval och fält som ska ignoreras när de är okända är det utrymme ett protokoll lämnar åt sin egen framtid.

Använd begreppet i ett beslut

Använd ”Vad är ett nätverksprotokoll?” 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

Ett nätverksprotokoll är det som gör båda ändarna utbytbara. Vilken klient som helst som har skrivits mot dokumentet kan tala med vilken server som helst som har skrivits mot det. En läsare kan alltså byta mjukvara, och en server kan skrivas om i ett annat språk, utan att den andra sidan märker det. Program som bara är överens för att de byggdes tillsammans kan inte ändras var för sig. För den som väljer hur en ny tjänst ska tala med omvärlden är det skälet till att ett etablerat protokoll nästan alltid vinner över ett eget: klienterna, biblioteken och felsökningsverktygen finns redan.

Protokollet avgör också mycket av säkerhetsläget innan någon kod har skrivits. Vilken sida som öppnar anslutningen avgör vad brandväggen måste släppa in: en server som väntar på klienter behöver en port som är öppen mot världen, medan ett program som bara ringer ut klarar sig utan. Vad meddelandena bär avgör vad som kan loggas, räknas eller läcka. En begäran som namnger en sökväg kan loggas per sökväg, och att dölja en läsares nätverksadress hindrar inte servern från att räkna förfrågningar. Ett protokoll som körs utan kryptering visar varje begäran för varje nätverk det passerar.

Till sist överlever protokoll de program som talar dem. Implementationer byts ut med några års mellanrum medan samtalet pågår i årtionden, och varje beteende som driftsatta program förlitar sig på blir en del av protokollet, vad dokumentet än säger. Internet Architecture Board har skrivit om båda halvorna av det problemet: misstag som tolereras tills de blir standard, och utrymme för tillväxt som ingen använder tills det har slutat fungera.

Så ser det ut i praktiken

Den här webbplatsen svarar på fyra protokoll, ur en enda mapp med filer på en enda server: HTTP, Gemini, Gopher och en onion-tjänst över Tor. Sidan beskriver Gopher som ett menyprotokoll från 1991 med ren TCP, ingen kryptering och inga skript, och visar en hel begäran skriven för hand:

```
printf '/contact\r\n' | nc engineer.company 70
```

Sidan kallar den raden hela protokollet. Under ytan skriver en driftsättning en mapp och fyra ytterdörrar läser den, så ingen av dem är en separat underhållen kopia som kan glida isär från de andra.

Brandväggsregler verifieras efter position snarare än förekomst, eftersom ett paketfilter utvärderar en ordnad lista och stannar vid första träffen. En av kontrollens regler gäller protokoll: det finns exakt en regel per port, och alltid med ett protokoll, eftersom en regel utan protokoll i tysthet vidgar attackytan. Den deklarerade publika ytan är en kort lista med fem portar som verifieras som helhet, så en port som dyker upp utan att vara deklarerad får kontrollen att fallera.

Den fjärde dörren är en onion-spegel, och dess protokoll formade både brandväggen och rapporteringen. Anonymitetsdaemonen når nätverket genom att bygga utgående kretsar, så ingenting ansluter inåt, ingen port öppnas och inget av brandväggsskikten har något att synkronisera. Gemini-förfrågningar och Gopher-anslutningar räknas per dag, utan adress och utan sökväg. Onion-tjänsten har ingen räknare och ingen flagga för att lägga till en, eftersom den här webbplatsen medvetet avstår från att räkna den publiken.

Var det går fel

Ett tolererat misstag blir standard. Robusthetsprincipen, var konservativ i det du skickar och liberal i det du tar emot, var tänkt att hjälpa program att fungera ihop. Tillämpad utan gräns låter den en bristfällig implementation släppas och spridas. Nästa implementation måste acceptera samma brist för att fungera med den, och bristen blir en de facto-standard som allt nytt måste vara kompatibelt med, bugg för bugg. RFC 9413 säger det rakt ut: de driftsatta implementationerna blir ett konformitetstest som kan ta över specifikationens roll. Avvisa det som specifikationen förbjuder, så tidigt i protokollets liv som möjligt, och gör felet synligt för någon som kan rätta det.

Två program är oense om var ett meddelande slutar. I en byteström ligger gränsen där protokollet säger att den ligger. Ett protokoll som erbjuder två sätt att uttrycka den, eller en parser som godtar en slarvig variant, låter en proxy och servern bakom den läsa samma byte som olika meddelanden. Det som den ena behandlar som slutet på en begäran behandlar den andra som en ny begäran, en som proxyn aldrig granskade. Tillåt en enda regel för inramning, avvisa allt som är tvetydigt och se till att varje led tolkar den på samma sätt.

En regel namnger en port men inget protokoll. Numret på en port säger ingenting förrän en transport är angiven, och beroende på verktyg kan en brandväggsregel som bara anger ett nummer öppna det för både TCP och UDP. Tjänsten som behövde det ena får det, och det andra står öppet för det som råkar lyssna där senare. Skriv in protokollet i varje regel och jämför hela den öppna ytan med en deklarerad lista.

Utrymmet för tillväxt slutar fungera. Utökningspunkter som ingen driftsatt avsändare använder rostar fast. Mottagare skrivs som avvisar det de aldrig har sett, och det första verkliga tillägget går sönder mot dem. Ju längre en sådan

mottagare är i drift, desto svårare blir den att rätta. Använd utökningspunkterna från början, även med värden som inte betyder något, så att toleransen prövas hela tiden i stället för att tas för given.

Läs mer

- NIST CSRC glossary: protocol — definitionerna som används i NIST:s säkerhetspublikationer, var och en med det dokument den kommer från.
- RFC 9293: Transmission Control Protocol — den gällande TCP-standarden, med byteströmmen som saknar gränser mellan meddelanden.
- RFC 768: User Datagram Protocol — hela UDP på tre sidor, skrivet 1980.
- RFC 1436: The Internet Gopher Protocol — en informativ beskrivning av ett helt applikationsprotokoll, med exempel på sessioner.
- RFC 9110: HTTP Semantics — vad förfrågningar, svar, metoder och statuskoder betyder, oberoende av vilken version som går över tråden.
- RFC 9413: Maintaining Robust Protocols — Internet Architecture Board om varför fel som tolereras biter sig fast.

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 ett nätverksprotokoll?” 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/protocol/