Martin Eriksson
Martin Rund

Anonyma P2P nätverk.

Abstrakt

Vi skall försöka beskriva hur ett anonymt peer-to-peer nätverk fungerar. Hur teorierna för hur det skall eller borde fungera och även försöka förklara hur en del implementationer använder sig av dessa teorier.

Abstract

We will try to describe how an anonymous peer-to-peer network work and to describe the theories that exists for anonymous peer-to-peer networks. We will also try to explain how some of existing implementation makes use of the theories.

Keywords: anonymous p2p, networking, p2p implementations, peer-to-peer.

Introduktion

Idag när Internetbrandbredden ökar hos hushållen, ökar också möjligheten att skicka stora mängder information mellan varandra. World Wide Web (WWW) är ett sätt för användaren att hämta information, men för att dela med sig av sina filer behöver man använda sig av något annat enkelt sätt. Det är detta peer-to-peer nätverk är till för, att kunna dela med sig information till andra och även hämta information från andra, en tvåvägskommunikation istället för den (till stor del) enkelriktade informationsflödet som WWW erbjuder. Peer-to-peer erbjuder alltså ett sätt att utbyta information snabbt och enkelt mellan ett flertal användare, egentligen ingen övre gräns (undre gräns är förstås 2).
Om nu alla kan delta och utbyta information i olika peer-to-peer nätverk, betyder det också att det går ta reda på vad de olika personerna som ingår i ett peer-to-peer nätverk tycker och tänker, detta genom att man vet vilken information som en person delar med sig. I ett land som censurerar information för folket, de vill förstås inte att olämplig information skickas runt, som de inte kan censurera. Om de som censurerar vet vem eller vilka som hämtar och delar med sig av denna för dem olämpliga informationen, skulle detta inte vara till fördel för folket. Det finns alltså ett behov för att kunna dela med sig information med varandra, där man inte behöver vara orolig att det går att spåra vart information kommer eller vart den skickas. Det är detta ett anonymt peer-to-peer nätverk skulle kunna åstadkomma.

Martin Eriksson
Martin Rund

Bakgrund

Anonymt peer-to-peer nätverk är på ingående, speciellt för vanliga användare som vill dölja sin identitet, vad det hämtar för information och vad de delar med sig. Vi skall försöka ge en överblick över hur anonyma peer-to-peer nätverk kan fungera.

Implementationer

Vi skall försöka att beskriva olika implementationer av anonyma peer-to-peer clienter, deras fördelar och nackdelar. Hur dessa implementationer verkligen är anonyma eller om det går att ta reda vem information kommer ifrån eller till vem man skickar information till.
Den primära skillnaden mellan ett anonymt och ett vanligt peer-to-peer nätverk ligger i de routing metoder som används i respektive nätverksarkitektur.
Ett ‘darknet’ är typiskt ett stängt litet privat p2p nätverk bestående av betrodda individer.

Freenet

Freenets plan är att bygga ett globalt darknet bestående av små nätverk som är sammankopplade, ungefär som Internet själv är uppbyggt.
Freenet är ett anpassningsbart peer-to-peer nätverk bestående av noder, som frågar varandra för att hämta och spara information, dessa noder har namn som är oberoende av vart de befinner sig. En nod håller reda sitt eget informationsutrymme, som den låter andra noder att skriva och läsa i. Varje nod håller även reda på en dynamiskt routing tabell som innehåller vägar till andra noder. Det är tänkt så att varje användare skall tillhandahålla noder, för att öka utrymmet (lagringskapaciteten) i hela freenet-nätverket.

Filnycklar

För att identifiera en fil i freenet används olika typer av nycklar. Den viktigaste nyckeln är Content Hash Key, CHK, vilken genereas genom att man hashar filens binära innehåll med SHA-1. Alla andra nycklar pekar på denna nyckel som identifierar filen.

En annan nyckel är Keyword Signed Key, KSK. Denna nyckel består av fri text, vars mening är att beskriva filens innehåll. Vanligt är att man beskriver en fil med många olika nyckelord som är hierarkiskt ordnade och åtskiljda med slashtecknet, jämför filsystem. För att komma åt innehållet i en fil som identifieras med hjälp av KSK, används två steg. Först beräknas hashvärdet, det vill säga CHK, av KSK:n och filen som motsvarar denna CHK hämtas. Då denna fil inte är den sökta filen utan innehållet endast är en pekare, ett CHK värde, till den sökta filen måste en ny hämtning ske innan den eftersökta filen kan läsas.
Det finns flera problem med KSK, varför den numera är föråldrat och därmed inte skall användas mer.

Martin Eriksson
Martin Rund

Ett av problemen med KSK är att namnrymden är platt och kollisioner därför kan inträffa om två användare använder samma KSK för två olika filer. Får att lösa detta problem infördes Signed Subspace Key, SSK. Med hjälp av SSK kan varje användare definiera en privat namnrymd och på så sätt undviks namnkollisioner. SSK fungerar som KSK med den skillnaden att ett prefix, som definierar en namnrymd, adderas till fritexten som beskriver filen.
För att använda SSK måste användaren skapa ett nyckelpar med hjälp av Diffie-Hellman. Den privata delen i nyckelparet används för att signera dokument och den publika delen för att verifiera signaturen. Prefixet som används i SSK nyckeln är hashen av den publika nyckeln vilket också definierar namnrymden. Då namnrymden definieras av hashen av den publika nyckeln innbär det att endast innehavaren av den privata nyckeln kan addera filer till namnrymden. För att göra det möjligt att verifiera signaturen, både för noder som lagrar filen och för clienter som använder filen, lagras den publika nyckeln okrypeterad i filen. Att den publika nyckeln är korrekt kan verifieras med hjälp av nyckelns hash, vilket är prefixet i SSK.

Kryptering av filer

Filer som lagras i Freenet är krypterade med en symmetrisk krypteringsnyckel. Denna nyckel måste vara känd av de klienter som vill använda filen samtidigt som den skall vara okänd för de noder som lagrar filen. Detta för att ägarna av noderna skall kunna förneka att de känner till filens innehåll. För att göra detta möjligt ingår den symmetriska nyckeln som en del i adressen till filen.

Ett exempel på en adress till en fil i Freenet är:
SSK@GB3wuHmtxN2wLc7g4y1ZVydkK6sOT-DuOsUo-eHK35w,c63EzO7uBEN0piUbHPkMcJYW7i7cOvG42CM3YDduXDs,AQABAAE/testinserts-3

Adressen består av:

Nyckeltyp:	SSK
Namnrymd:	GB3wuHmtxN2wLc7g4y1ZVydkK6sOT-DuOsUo-eHK35w
Krypteringsnyckel:	c63EzO7uBEN0piUbHPkMcJYW7i7cOvG42CM3YdduXDs
Krypteringsalternativ:	AQABAAE
Filbeskrivning:	/testinserts
Filversion:	-3

Lagring

I freenet kan man lägga till och hämta filer, men man kan aldrig ta bort en fil. För att lägga till en fil skickar man ett insert meddelande och för att hämta en fil skickas ett retrive meddelande. Borttagning av filer sköts däremot av noden själv, då lagringsutruymme behövs för nya filer. Detta sker enligt Least Recent Used, LRU, det vill säga, blir det utrymmesbrist på en nod raderas den fil som har det största tidsspannet sedan den lästes senast.

Då noder kan välja att casha filer som efterfrågas kan man heller aldrig avgöra på hur många noder en fil finns lagrad. För att en fil helt skall raderas från Freenet krävs att samtliga noder som lagrar en fil, måste radera filen i enlighet med LRU.
Radering enligt LRU i kombination med cashning av filer innebär att filer flyttas dynamiskt i nätet till platser som ligger ”nära” clienter som använder en fil mer frekvent. Att en nod kan välja att casha eller inte att casha en fil gör att man aldrig med säkerhet kan säga exakt var en fil finns lagrad även om man gör upprepade läsningar av en fil.
Vidare finns ännu ingen möjlighet att söka efter filer i Freenet.

Routing

En nod i Freenet kommunicerar bara med sina logiskt närmsta grannar. Om ett meddelande måste skickas till någon annan nod, sker detta alltid genom grannnoden. Två noder som inte är grannar kommunicerar aldrig direkt med varandra. Detta gör att man aldrig kan avgöra om filen man frågar efter kommer från grannen eller grannens granne osv. vilket säkerställer att man aldrig kan ta reda på var filen kommer ifrån.
För att en nod skall kunna göra inteligenta val av vilken granne som en förfrågan skall skickas till, har varje nod en routingtabell som kontinuerligt uppdateras. Denna routingtabell innehåller grannar och CHK. En nod läggs in i routingtabellen med en CHK, om en förfrågan med aktuell CHK ger träff hos denna nod. Då en nod skall skicka ut en förfrågan om en CHK till en annan nod, väljs den nod ur routingtabellen som har den lexikalt närmaste nyckeln, i förhållande till den efterfrågade CHK:n.
Om det inte skulle vara möjligt att skicka en förfrågan till den nod med den närmsta nyckeln skickas förfrågan till den nod som har den lexikalt näst närmaste nyckeln och så vidare tills dess att det inte finns någon mer att skicka till. Orsakerna till att en förfrågan inte kan skickas till en viss nod beskrivs nedan. Då en nod får ett positivt svar från en annan nod uppdateras routingtabellen med den nya informationen.
Varje meddelande har en hops-to-live som räknas ned varje gång ett meddelandet skickas vidare. När hops-to-live har kommit till 0 och den efterfrågarde filen inte har påträffats returneras en file-not-found meddelande som går tillbaka till den nod som initierade frågan, via alla noder som frågan har gått genom. Hops-to-live kan jämföras med IP:s time-to-live och förhindrar att ett meddelande skickas till noder i all oändlighet.
Varje meddelande har även ett unikt identifikationsnummer vilket gör att en nod kan identifiera ett meddelande när svar kommer tillbaka. För att detta skall vara möjligt måste varje nod hålla reda på vilka meddelanden som är aktiva, dvs skickade förfrågningar där svar ännu inte kommit tillbaka. Detta görs för att loopar skall undviks då en fråga skickas. När en nod får ett meddelande kontrolleras först om det aktuella meddelandet är aktivt på noden, är det aktivt returneras ett meddelande som säger detta och den frågande noden får skicka meddelandet till någon annan. Finns ingen annan att skicka till returnerar denna nod file-not-found uppströms.
Det faktum att förfrågningar skickas till noder som har CHK nycklar i routingtabeller som ligger nära den eftersökta CHK:n, innebär det att över tiden kommer noder specialicera sig på närliggande CHK:er. Denna lokalitet innebär att förfrågningar kan hanteras mer effektivt efter

viss tid, då man med större säkerhet kan skicka ett meddelande till rätt nod på första försöket. En nod lär sig med tiden hur Freenet ser ut, varför en nod blir effektivare ju längre den har varit aktiv. Då en nod startas upp för första gången, har den ingen uppfattning om hur Freenet ser ut och vilka noder som lagrar vilka filnycklar. Detta innebär att förfrågningar kommer att ske till slumpmässigt valda noder. Då varje nod slumpar detta på olika sätt kan man ej förutse hur filer kommer att lagras i nätet, varför detta sker slumpartat. Då filer efterfrågas olika mycket från olika klienter, kan klustrade filer återfinnas på flera olika noder i nätet.

Retrive och insert

Det finns två primitiver i Freenet, retrive och insert. Detta är de enda operationer en klient kan utföra. För att hämta en fil skickar klienten ett retrivemeddelande till närmasta nod, vilken oftast körs på samma maskin som klienten. Då en nod får ett retrivemeddelande används ovan beskrivna routingmetod för att hitta och leverera filen.

Då en klient vill spara en fil under Freenet skickas ett insertmeddelande. Detta meddelande hanteras på samma sätt som ett retrivemedelande med den skillnaden att filen sparas endast om den inte hittas. Om en fil med identisk nyckel redan finns på freenet, kommer den att hittas eftersom ett insertmeddelande och ett retrivemeddelande hanteras och routas på samma sätt i systemet. Detta innebär också att en ny fil kommer att lagras på en nod där filer med lexikalt närliggande nycklar finns lagrade. Detta bidrar till att lokaliteten, vad gäller filnycklar, bibehålls även för helt nya filer i systemet.

Verktyg/Klienter

Freenet är ett underliggande nätverk som olika verktyg kan använda för att spara och hämta filer. Det finns dels specialskrivna klienter för att administrera en freesite. Freesite är en webbplats under freenet, vilken man kan använda sin befintliga webbläsare via Fred, Freenet referens daemon, för att läsa.
Det finns även klienter för att direkt ladda upp och ner filer på Freenet, dessa klienter fungerar på samma sätt som vanliga p2p-klienter med den skillnaden att filerna laddas upp och ner mot freenet istället för mot en adresserad host.
Utöver dessa typer av klienter finns även fria bibliotek som kan användas för att skriva egna tillämpningar som använder Freenet.

Winny

Detta är en japanskt peer-to-peer nätverk för att dela information, som troligtvis är baserad på freenets arkitektur. Det sägs att det kan hålla användändarnas identitet hemlig och att den inte skall kunna spåras. Den största skillnaden mellan Winny och Freenet är utvecklingsspråket, där Freenet är skrivit i Java och Winny i C++.
Winny är inte ett opensource projekt och hur systemet är uppbyggt och fungerar i detalj är inte allmänt känt. Det har via den japanska versionen av Ebay sålts ett dokument som beskriver hur Winny är uppbyggt och fungerar. (vilket vi inte har köpt!)

Martin Eriksson
Martin Rund

Lagring

Winny hämtar hem och sprider data automatiskt, den skickar dock bara ut information som användaren har valt att dela ut.
Genom att man skriver in klusterord, så kommer nätverket (Winny) att sammankoppla klienter som har använt sig av likartade klusterord, på så sätt är inte alla sammankopplade med alla, vilket hade ökat overheaden i trafiken ordentligt. Om man till exempel skulle skriva ”mp3” bland sina klusterord, skulle Winny koppla samman dig med andra som har mp3-filer och troligtvis har de personerna en massa mp3-filer.

Säkerhet

Genom att inget vet vem som har information eller om den man skickar till verkligen skall ha den, går det inte att säga att den som hämtar detta från mig, det är han som skall ha filen.
Winny har även ett inbyggt forumliknande verktyg, där man kan skriva in information om en eventuell programvara man själv delar ut. Ett inlägg i forumet är inte anonymt, varav att man då vet att just denna IP-adress delar ut just den filen som foruminlägget handlar om. När man hämtar filen, kan man se att det kommer paket från samma IP-adress, då vet man nästan säkert att just innehavaren av den IP-adressen har filen det handlar om.

I2P

I2P är en förkortning av ”Invisible Internet Project”, som är ett pseudoanonymt virtuellt nätverk. Detta är egentligen ingen applikation i säg, utan är bara ett nätverkslager som andra applikationer kan använda sig av för att kommunicera anonymt och säkert men varandra.

Tunnlar

För att inte någon skall veta till vem eller från vem ett paket kommer, skickas paket genom tunnlar som skapats av I2P, varje nod (klient) som sitter mellan den som skickar paketet och den som skall ha paketet, tar emot ett medelande (paket), eventuellt flera, krypterar paketen och skickar vidare det till nästa klient, som upprepar processen tills att paketen har hamnat hos den som skulle ha det. Varje paket är krypterade i flera omgångar, vilket gör att bara den rätta mottagaren kan dekryptera paketet.
I2P profilerar hela tiden klienterna för att bestämma hur tillförlitliga och belastade de är, detta för att vid behov, så vet I2P vilka klienter som är lämpligast att använda som tunnlar. Det är då även möjligt att välja en annan väg som inte har lika belastade klienter.

Nätverksidenitet

För att hålla reda på vem som är vem när alla är anonyma, krävs ett sätt tabell eller databas för att veta vem tunnlarna skall kopplas upp till och vart paket skall skickas när de ankommer. Strukturen på informationen i denna tabell innehåller information för hur man skall skicka information säkert till nästa router, vilket innehåller följande:
Identitet, 2048 bitars ElGamal publiknyckel, 1024 bitars DSA publiknyckel och ett certifikat.

Martin Eriksson
Martin Rund

IP-adressen och porten.
När information skapades (utlämnades från klienten).
Signatur, skapad av den inkluderade 1024 bitars DSA nyckeln.
Nyckeln för denna information är en SHA256 hashning av routerns egna identitet.

Protokoll

Kommunikationen mellan routrar (klienter), behöver ett protokoll för tillförlitlighet och det skall inte kunna avlyssnas av utomstående, det behöver också kunna säkerställa att den man skickar medelande till verkligen är den som skall ha det. I början användes ett TCP baserat protokoll för att uppnå detta. Nu används SSU istället (Secure Semireliable UDP),
Från SSU specifikationen:
The goal of this protocol is to provide secure, authenticated, semireliable, and unordered message delivery, exposing only a minimal amount of data easily discernible to third parties. It should support high degree communication as well as TCP-friendly congestion control, and may include PMTU detection. It should be capable of efficiently moving bulk data at rates sufficient for home users. In addition, it should support techniques for addressing network obstacles, like most NATs or firewalls.

Garlic Meddelande

Garlic meddelande är en vidarutveckling av ”onion”-lager krypering, vilket tillåtar att ett meddelande kan innehålla flera andra medelande vid sidan av sin egna information för vart meddelandet skall skickas. Detta gör så att man kan skicka information krypterad som egentligen skulle ha skickats i klartext, vilket betyder att enbart mottagaren får reda på klartexten. Ett exemplet är när en klient vill skicka ett medelande, sändarens router komma då att lägga ihop meddelandet tillsammans med andra meddelande till ett ”garlic”-medelande, kryptera detta med en 2048 bitars publik ElGamal nyckel som mottagaren har och sedan skicka iväg ”garlic”-medelandet genom rätt tunnel.

Säkerhet

Ett antal krypteringstekniker används för att tillsammans ge ett säkert lager som kan motstå attacker från icke behöriga personer. Detta görs på flera nivår:
Mellan routar, används SSU (Secure Semireliable UDP) vilket krypterar varje paket med AES algoritmen i CBC läge med en 256 bitars nyckel med unika IV's och MAC (HMAC-MD5-128) denna nyckel skapas och utbytes genom att en 2048 bitars Diffie-Hellman utbytes algoritm används. Tunnlarna i sin tur använder sig av ett eget lager av AES256 i CBC läge, plus ett SHA256 hash värde för att verifiera att paketet är det rätta. Nätverket verkar att vara så säkert att ingen utomstående kan avlyssna trafiken, i alla fall inte förstå vad som skickas. Men som sagt är I2P bara ett nätverkslager för att skicka informationen, det betyder inte att

Martin Eriksson
Martin Rund

programvaran som användersig av I2P är säkert bara för att det just använder sig av I2P. Det kan inte vara starkare än den svagaste länken.

Martin Eriksson
Martin Rund

Diskussion

Fördelar respektive Nackdelar, det finns många som tycker att information skall få flöda fritt mellan personer, men all information är inte fri. Vi har ju lagar som gör att man inte kan ta någons annans arbete och sälja utan tillstånd från upphovsmannen. Samma sak gäller datorprogram och annan digital information. Vissa försöker att likna peer-to-peer nätverk stölder ur en affär, att kopiera ett program utan tillstånd är som att gå in i en affär och ta en TV utan att betala. Visst kan det ses så, fast det finns en väsentlig skillnad, för att bygga en TV krävs material, men genom att kopiera digital information står den som kopierar information för materialet själv. Om man då tittar på materialkostnaden för en TV och jämför detta med vad TVn kostar i affären, hur stor del av kostande går tillbaka till den personen eller företaget som designade TVn (elektronik och utseende). Denna kostnad, den kostnad som går tillbaka till upphovsmannen, det är det som TVn borde kosta om man står för materialet själv. Samma tankesätt skulle man kunna använda för digital information. Om man jämför med en cd-skiva, ett album. Om det är möjligt att hämta informationen och betala den verkliga kostnaden, till upphovsmannen för arbetet denna lagt ner. Detta hade varit bra för alla (förutom alla mellanhänder som inte längre får in några pengar). Genom att dessa mellanhänder idag gör allt för att förhindra kopiering av upphovsrättsskyddat material uppstår en efterfråga på anonyma peer-to-peer nätverk, vilket direkt motverkar det köpsamhälle vi faktiskt lever i idag. Konsumenterna (peer-to-peer användare) går åt ena hållet och mellanhänderna går åt andra, de som blir mest lidande är då de som tillverkar och konstruerar information som går att kopiera på digital väg.

Referenser

[] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. , *Freenet: A Distributed Anonymous Information, Storage and Retrieval System*
[] Freenet 2006, *http://www.anonymous-p2p.net/*
[] Wikipedia 2006, *http://en.wikipedia.org/wiki/Freenet*