Kryptografi & Anonymitet:

Etik och juridik i konflikt

Victor Rudolfsson
Høgskolen i Gjøvik
Gjøvik, Norge

I dagens samhälle advokerar ett ökande antal organisationer och regimer för en centraliserad lösning av förvaring och indexering av sin egen och befolkningens data, genom föreslagna och införda lagar[10] och policies ämnade att öka brottsbekämpning.
Samtidigt advokerar ett ökande antal individer och kollektiv ofta benämnda *cypherpunks* eller *kryptoanarkister* för en mer decentralizerad lösning för att säkra information (och privatliv) baserat på kryptografiska algoritmer.

En fråga vi tvingas ställa oss är varesig vår data någonsin är säker, och vem eller vilka vi vill ge ansvaret att säkra vår data. Men än viktigare är kanske frågan om varesig en centraliserad lösning är att föredra över en decentraliserad lösning till att börja med. Flera av de lagar ämnade att öka brottsbekämpning genom att samla in, förvara och indexera data sägs vara ämnade särskilt för att komma åt de som begår andra slags brott, så som distribuering av barnpornografi, terrorism och ekonomiskt relaterade brott.

Men när majoriteten av en befolkning är övervakad näst intill (om inte) konstant, och deras kommunikation därefter sparad och indexerad, kan denna information verkligen användas för att bidra till brottsbekämpning I den mån det framställs vara ämnade för, när de som de sägs vara ämnade att övervaka utan större svårigheter kan undkomma övervakning genom diverse kryptografiska algoritmer och protokoll för att säkra sin kommunikation? Och kommer det isådanafall ställa till ytterligare problem vi tvingas lösa för att uppnå den önskade nivån av rättsamhälle?
Det finns många olika tekniker som kan tillämpas för att kryptera överföring idag, och än fler alternativ till de olika teknikerna (*IPSec*, *SSL & PPTP* för VPN, *SSL,* eller *SSL/TLS, TOR*[4], och *I2P,* etc*),* men även jag kommer i huvudsak rikta in mig på att kortfattat beskriva hur några av de mest vanliga krypteringsprotokollen för datakommunikation fungerar, och försöka ta reda på huruvida dessa kan användas för att säkra kommunikation på ett decentraliserat vis, och om en centraliserad lösning isådanafall kommer att behövas, samt om en decentraliserad och centraliserad lösning kan komma att hamna i konflikt.

Data som överförs via internet skickas med hjälp av ett eller flera av de tillgängliga protokollen för överföring – de som tas upp här är huvudsakligen TCP[1] och UDP[2], eftersom det är de som används av TOR, SSL/TLS, och OpenVPN vilka är de protokoll jag valt att fokusera på. *Se bilaga nr 1 för en illustration av ett TCP paket.*

Eftersom TCP/IP inte är ett enda protokoll, utan en samling av flera protokoll, är de olika protokollens ansvarsområden uppdelade i olika *lager*[3]. TCP har fyra olika lager (av vilka var och ett kan länkas samman med ett eller flera lager i OSI-modellen), och eftersom ansvaret för kryptering tilldelas *transport* lagret i ett TCP paket, är det där de flesta av protokollen som nämns har sin plats.

SecureSocketLayer/TransferLayerSecurity

Ett av de ansvarsområden som tilldelas transportprotokollet är att upprätta en anslutning, vilket är varför det är särskilt viktigt att nämna att det är i detta lager SSL/TLS protokollet har sin plats. Det innebär nämligen att såvida inte säkerhetskontrollerna lyckas så upprättas inte en anslutning. [5a]

Processen för SSL/TLS protokollet består av två olika steg, det första steger är *autentisering*, och det andra är *kryptering/dekryptering.* Eftersom SSL/TLS är baserat på *certifikat*, måste båda parter bekräfta sin identitet för varandra genom att uppvisa digitalt signerade certifikat och validera dessa med certifikatutfärdaren innan autentiseringssteget är slutfört. Om detta misslyckas, upprättas ingen anslutning.

Under steg två använder sig SSL/TLS utav två nycklar – en publik och en privat. Den publika nyckeln används för att enkryptera, medans den privata nyckeln används för att dekryptera. Båda parter måste dela sin publika nyckel med den andra parten, och använder sedan de genererade privata nycklarna för att dekryptera datan de mottager när överföringen påbörjats. [5b]
Till skillnad mot SSL/TLS så kräver vanlig SSL[6] *endast att en part (servern) bekräftar sin identitet. Se bilaga 2 för en demonstration av hur ett TCP paket med SSL kan se ut.*

OpenVPN

OpenVPN, som betyder open-source Virtual Private Network, fungerar genom att skapa en krypterad tunnel av datapaket (ofta UDP, men

kan även använda sig av TCP paket) mellan två punkter. Kortfattat innebär detta att OpenVPN upprättar en ström av krypterade datapaket mellan punkt A och punkt B, och förlitar sig helt på OpenSSL för kryptering, och kan därav använda sig av antingen certifikat genom en speciell implementering av SSL/TLS, eller på förhand utdelade statiska nycklar. En än mer föränklar förklaring vore att data överförd med hjälp av OpenVPN är inkapslad och krypterad inuti en krypterad tunnel av datapaket. [7]

Se bilaga 3 för en demonstration av ett UDP paket I en krypterad UDP tunnel kan se ut vid användning av OpenVPN

The Onion Router

TOR, som står för '*The Onion Router*', är ett distribuerat nätverk ämnat för anonymisering på nätet. Detta fungerar genom att kapsla in data i flera lager kryptering, och slumpmässigt välja en väg från punkt A till punkt B genom nätverket i vilket ingen nod vet mer än vilken den föregående noden var, och vilken nästa nod på vägen till destinationen är. Var nod under vägen dekrypterar det översta lagret, innan paketet skickas vidare, och därav analogin till *onion (lök)* i namnet.

För kryptering använder sig TOR utav en *ström chiffer* (128-bitar AES i counter-läge), Diffie-Hellman protokollet, en publik chiffer (*1024-bitars RSA)* samt en hash funktion (*SHA1*). [8]
Se bilaga 4 för en demonstration av hur ett datapaket kan se ut när TOR används

Eftersom alla dessa metoder bidrar till en komplicerad dekrypterinsprocess – SSL kräver nycklar för dekryptering, medans TOR bidrar till att varje paket är krypterat i flera lager – innebär det att det inte är resursvänligt överhuvudtaget att försöka dekryptera dem genom att försöka gissa sig till ursprungsdatan, och för att lyckas dekryptera dem tvingas man mer eller mindre se på protokollens svagheter.

TOR har en relativt uppenbar svaghet: eftersom varje nod dekrypterar ett lager, innebär detta att den sista noden på vägen till destinationen dekrypterar det sista lagret, vilket innebär att såvida inte en punk-till-punkt kryptering som TLS eller SSL/TLS används så kommer datatrafiken att vara okrypterad det allra sista steget. Därför bör det per definition gå att avlyssna den sista noden, *exit node*, för att komma åt det dekrypterade innehållet, och kanske är avlyssning av slutnoder en av få, om inte den enda, effektiva lösningen för att avlyssna trafik I TOR nätverket [11].

För SSL å andra sidan ligger svagheten snarare hos användaren – eftersom certifikat vars nycklar inte ser ut som de bör allt som oftast resulterar i en varning i användarens webbläsare, vilket frekvent sker då servrar använder sig av utdaterade certifikat, accepterar användare ofta dessa varningar och tar dem inte på allvar. Det innebär även att en tredje part skulle kunna agera som en slags osynlig relä mellan server och användare, och uppvisa ett fabrikerat certifikat i hopp om att användaren inte tar varningen på allvar och initierar uppkopplingen oavsett, vilket i slutändan leder till att tredje parten accepteras som en relä mellan användare och server och tillåts dekryptera överföringen.

SSL är dock menat för att vara motståndskraftigt mot brute-force attacker, och gör det mycket svårt att dekryptera datan utan tillgång till nycklar. TOR, å andra sidan, är resistant mot brute-force attacker på grund av de flera lager av kryptering var datapaket kapslas in i. Både SSL och TOR har svagheter som mer eller mindre kräver en attack mot kommunikationen, servern eller klienten, och inte själva datan i sig. Däremot kan TOR i framtiden utsättas för avlyssning då slutnoder som befinner sig inom EU kan tvingas lagra den dekrypterade kommunikationen enligt exempelvis datalagringsdirektivet, men för närvarande ligger TORs svaghet i varesig slutnoden är pålitlig eller inte.

Även om dessa protokoll är användbara för kryptering och för att säkra kommunikation för de som känner behov för det, sägs det att vilket verktyg som helst kan bli ett vapen i fel händer – men innebär det att alla som använder verktyget bör misstänkas göra det i fel syfte, och är det etiskt rätt att avlyssna TOR's slutnoder, eller förfalska SSL certifikat och lura användare, för att försäkra sig om att ingen kommer undan det syfte som dessa lagar införts för? Eller är nästa steg att försäkra oss om att dessa föreslagna och/eller introducerade lagar kan verkställas i det syfte de var menade för, oavsett? Behöver vi tänka över vad vår etik verkligen innebär, eller är dessa lagar[9][10] etiskt definierade – och i så fall, justifierar det användandet av dessa tekniker för att verkställa dem?

Källförteckning

1. Structure of a TCP packet: RFC 793: Transmission control protocol, September 1981, section 3.1 page 15 (http://tools.ietf.org/html/rfc793),
2. Structure of UDP packet: RFC 768: User Datagram Protocol, 28 August 1980, http://tools.ietf.org/html/rfc768
3. Layers of TCP/IP: The TCP/IP model `http://technet.microsoft.com/en-us/library/cc786900(v=ws.10).aspx`
4. The Onion Router: www.torproject.org
5.
   a. What is TLS/SSL? http://technet.microsoft.com/en-us/library/cc784450(WS.10).aspx
   b. Authentication and data exchange: `http://technet.microsoft.com/en-us/library/cc783349(v=ws.10).aspx#w2k3tr_schan_how_hkrr`

6. The SSL Protocol Version 3.0: RFC6101, August 2011 (http://tools.ietf.org/html/rfc6101)
7. OpenVPN cryptographic layer: http://openvpn.net/index.php/open-source/documentation/security-overview.html
8. TOR Protocol specification: https://gitweb.torproject.org/torspec.git/blob/HEAD:/torspec.txt
9. Anti Counterfeiting Trade Agreement, draft from April 2010, http://trade.ec.europa.eu/doclib/docs/2010/april/tradoc_146029.pdf
10. Directive 2006/24/EC ("Data Retention Directive"), `http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2006:105:0054:0063:EN:PDF`
11. "*Rogue Nodes Turn Tor Anonymizer Into Eavesdropper's Paradise*", Wired Magazine, 2007-09-10, `http://www.wired.com/politics/security/news/2007/09/embassy_hacks`

Bilagor

1 – *Ett TCP paket innehållande meddelandet"o hai" skicket över* ***I****nternet* ***R****elay* ***C****hat*

```
▸ Frame 233: 94 bytes on wire (752 bits), 94 bytes captured (752 bits) on interface 0
▸ Ethernet II, Src: HonHaiPr_af:d1:34 (90:00:4e:af:d1:34), Dst: JensenSc_08:9f:14 (34:21:09:08:9f:14)
▸ Internet Protocol Version 4, Src: 192.168.0.107 (192.168.0.107), Dst: 195.140.202.142 (195.140.202.142)
▸ Transmission Control Protocol, Src Port: 56799 (56799), Dst Port: ircd (6667), Seq: 201, Ack: 7887, Len: 28
▾ Internet Relay Chat
  ▾ Request: PRIVMSG #pandaparty :o hai
      Command: PRIVMSG
    ▾ Command parameters
        Parameter: #pandaparty
      Trailer: o hai
```

2 – *Ett TCP paket med SSL aktiverat, vars innehåll iövrigt är identiskt med paketet ovan*

3 – *Ett UDP paket i en UDP tunnel med OpenVPN*

```
▸ Frame 606: 135 bytes on wire (1080 bits), 135 bytes captured (1080 bits) on interface 0
▸ Ethernet II, Src: HonHaiPr_af:d1:34 (90:00:4e:af:d1:34), Dst: JensenSc_08:9f:14 (34:21:09:08:9f:14)
▸ Internet Protocol Version 4, Src: 192.168.0.107 (192.168.0.107), Dst: 195.140.202.142 (195.140.202.142)
▸ Transmission Control Protocol, Src Port: 55979 (55979), Dst Port: 6697 (6697), Seq: 1156, Ack: 10522, Len: 69
▾ Data (69 bytes)
    Data: 170301004014a6578bccdf722c6d6777da614df8a29c444b...
    [Length: 69]
```

4 – *Ett paket skickat via TOR nätverket*

```
▸ Frame 2518: 135 bytes on wire (1080 bits), 135 bytes captured (1080 bits) on interface 0
▸ Ethernet II, Src: HonHaiPr_af:d1:34 (90:00:4e:af:d1:34), Dst: JensenSc_08:9f:14 (34:21:09:08:9f:14)
▸ Internet Protocol Version 4, Src: 192.168.0.107 (192.168.0.107), Dst: 46.21.99.25 (46.21.99.25)
▸ User Datagram Protocol, Src Port: 37470 (37470), Dst Port: openvpn (1194)
▾ Data (93 bytes)
    Data: 30db7780ebc6512955edc08373f698cdce1519341626897e...
    [Length: 93]
```

```
▸ Frame 4517: 652 bytes on wire (5216 bits), 652 bytes captured (5216 bits) on interface 0
▸ Ethernet II, Src: HonHaiPr_af:d1:34 (90:00:4e:af:d1:34), Dst: JensenSc_08:9f:14 (34:21:09:08:9f:14)
▸ Internet Protocol Version 4, Src: 192.168.0.107 (192.168.0.107), Dst: 50.7.161.218 (50.7.161.218)
▸ Transmission Control Protocol, Src Port: 56044 (56044), Dst Port: https (443), Seq: 26275, Ack: 44358, Len: 586
▾ Secure Sockets Layer
  ▸ TLSv1 Record Layer: Application Data Protocol: http
  ▾ TLSv1 Record Layer: Application Data Protocol: http
      Content Type: Application Data (23)
      Version: TLS 1.0 (0x0301)
      Length: 544
      Encrypted Application Data: af4a1cb04c27e31a3663638fbd7acca2afb6653f74c1963e...
```