Introduction slide

In today's society, a growing number of organizations and governments advocate a centralized approach to safeguarding its own and its people's data, by introducing and enforcing laws and policies supposed to aid law enforcement.

At the same time, a growing number of people and collectives sometimes referred to as "*cypherpunks*" or "*cryptoanarchists"* advocate a decentralized solution to the safekeeping of data (and privacy) based on cryptographic algorithms and protocols.

A question we must ask ourselves is whether our data is ever safe, and who we wish to be responsible for keeping it safe. But more importantly, whether or not a centralized approach is useful to begin with. Many of these laws meant to harvest, store and index data are said to be meant for enforcing other laws, such as stopping the distribution of child pornography, finding and arresting terrorists, and stopping financial fraud.

However, when a majority of a country's population is monitored most of, or all the time, and their communications stored and indexed, can this information really be used to aid law enforcement when those who these laws are meant to stop can easily avoid them by utilizing transfer protocols based on cryptographic algorithms to secure communication end-to-end? And do such protocols actually pose a problem for law enforcement?

Origin of the Cypherpunks term and culture

Until the beginning of the 70s encryption was almost only used by for military purposes, however, by the end of the decade two publications in particular brought encryption to the public eye. These were *DES (Data Encryption Standard)* and the *Diffie-Hellman key exchange.*

This lead to an open discussion about cryptography and speculations such as if it could be used by criminals to giver their tracks, and thus if encryption should be limited, or encouraged in order to protect privacy.

These discussions sparked an online group referred to as *cypherpunks,* consisting of people interested in cryptography and the protection of privacy which by the end of the 80s had started to seem more like a movement than just an online group. Their main source of communication was originall the *cypherpunk mailing list* and its forks and successors.

The cypherpunks were the origin of cryptoanarchism, and were the ones to come up with the word to begin with, and originally described it as a type of anarcho-capitalism, since they realized it would be possible to create a virtual currency to allow for trade and commerce by using cryptographic algorithms.

Cryptoanarchism and Cypherpunks

Crypto-anarchism, is thus more politically focused ideology, and a branch of anarchist philosophy that focuses on the use of technology to protect privacy, and gain autonomy from government and in some cases corporate actors in communication, trade and information security. Cryptoanarchists make use of cryptographic software to protect their privacy, and believe that cryptography is the key to protecting personal privacy and defending oneself from censorship, rather than political action.

In many ways, cypherpunks are therefore closely related to cryptoanarchists, albeit not entirely the same. As written in the *cypherpunks manifesto* by Eric Hughes in 1993:

"*We the Cypherpunks are dedicated to building anonymous systems. We are defending our privacy with cryptography, with anonymous mail forwarding systems, with digital signatures, and with electronic money.*
Cypherpunks write code. We know that someone has to write software to defend privacy, and since we can't get privacy unless we all do, we're going to write it. We publish our code so that our fellow Cypherpunks may practice and play with it. Our code is free for all to use, worldwide. We don't much care if you don't approve of the software we write. We know that software can't be destroyed and that a widely dispersed system can't be shut down."

While the main source of communication for the original cypherpunks movement is today almost dead (the cypherpunk mailinglist), there are other mailing lists that have taken over as successors – but more active moments today are closely related to the cryptoanarchist philosophy (*Telecomix, is one such movement for example*).

Privacy and Cryptography

Most cypherpunks argued that for freedom of expression to truly thrive, one must be allowed privacy. If you say something to somebody, and you want it to be heard only by that person, encryption is not only a useful tool but a required tool in today's electronic era. In past times this could be compared to whispering something to somebody, or sending letters with codewords, or similar acts – but with the technology of today, that could be compared to encryption.
If you cannot express yourself without worrying about being heard by the wrong person, can you truly express yourself freely?

With data retention laws such as the Data Retention Directive in Europe, the proposed "*OzLog*" in Australia, or the proposed *SAFETY* law in the United States being proposed in different forms all over the world, one must ask how much privacy one is really entitled to. Another important question to ask may be if such approaches are useful in reality, without also prohibiting the anonymity and plausible deniability provided by the use of different kinds of encryption – because if a majority of the people is monitored most of the time, will it help stopping those who have something to hide, if they can easily hide? And if so – will these laws affect the average persons right to privacy more than they will actually aid in reaching the intended purpose of aiding law enforcement? What use is it to store information, if it cannot be read?

There are many different techniques to encrypt data transmission, and there are several alternatives providing different kinds of security as well (*IPSec*, *SSL* & *PPTP* for VPN, *SSL* or *SSL/TLS*, TOR or I2P and so on), but I'm going to briefly explain how a few of the most commonly used encryption protocols for data communication work, and find out whether or not communication can be safeguarded in a decentralized manner, if a centralized approach is required and if these two approaches conflict with each other.

Data sent over the internet are transmitted using one of several available protocols. The ones brought up in this presentation are TCP and UDP, both part of the TCP/IP model, since those are the ones used by TOR, SSL/TLS and OpenVPN which are the protocols I've chosen to focus on.

TCP/IP

Because TCP/IP is not a single protocol, but rather a suite of protocols, the responsibilities of these protocols are organized into different *layers* . TCP has four different layers (each of which correspond to one or several layers in the OSI model).

These layers have different responsibilities in data transmission. The top layer is called the *application layer,* and this layer is responsible for communication on a process to process basis. For example, this is where the FTP, or HTTP protocol operates – the communication between an FTP client and an FTP server, or a web browser and the web server, is a responsibility assigned to this layer.

Below the application layer is the *transport layer*. This is the layer I'm going to focus most on, because this layer is responsible for host to host communication – your computer communicating with a server, for example – and encryption is a responsibility of this layer.

The layer below the transport layer is the *internet layer*, which is responsible for interlinked networking – communication between different networks through the use of gateways.

The bottom layer is called the *link layer,* or *ethernet,* and is responsible for establishing communication on the local network.

Because encryption is a responsibility assigned to the transport layer of a TCP packet, that's where the protocols we're going to describe very briefly resides.

No encryption provided by default

By default, no encryption is provided at all. This means that most messages sent will be sent in *cleartext* and therefore can be sniffed and read easily. However, by using a protocol such as SSL, or Open VPN, or even both in combination, information sent becomes harder to read even if it's picked up. On the top picture data is sent using absolutely no encryption, and as you can see it's easily readable.

On the bottom picture, the exact same message is sent to the exact same destination, but using SSL. Because establishing host to host communication is one of the things that the transport layer is responsible for, it's important to mention that this is where encryption protocols such as SSL/TLS resides: Because if the security checks are not passed, connection will not be established between the hosts.

SSL

The SSL/TLS procedure consists of two steps, the first one being *authentication* and the second one being *encryption.* Because SSL/TLS is based on the idea of using *certificates*, both sides have to prove their identity to one another by showing their digitally signed certificates and validating these with the certificate authority before the authentication process is complete. If this fails, a connection will not be established.

During the next phase, this protocol makes use of two keys – a public and a private one. The public key is used to encrypt, whereas the private key is used to decrypt. Both clients share

their public key with one another, and use their private keys to decrypt the data received once transmission has begun.
In contrast to SSL/TLS, regular SSL only requires one side (the server) to verify its identity.

The picture illustrates an SSL handshake between a client and a server.

VPN

A VPN is a different type of concept from SSL, and can be used together with SSL. VPN stands for *Virtual Private Network,* and as the name implies, establishes a private network between the clients and the server. It allows a server to act as a gateway through which all data to and from the client will pass, much like your modem and router would normally act on a local network – but the connection between the client and the server usually occurs over the internet, using an encrypted tunnel.

This means that for a website or a server a client is connected to, the traffic will seem to originate from the VPN server, which acts as an intermediate between the source and the destination, and it is in this that the anonymity provided by VPN lies – the source is never visible to the destination server.

There are multiple different protocols available for VPNs, such as IPSec, PPTP, Open VPN, to name a few. Because I don't have time to go through all of them, I'll focus on Open VPN.

The way OpenVPN works, is by creating an encrypted tunnel (usually UDP, but can also run over TCP) between two points. Simply put, Open VPN sends a stream of encrypted packets from point A to point B, using Open SSL for encryption and either certificates through a custom SSL/TLS implementation, or a pre-shared static key. To put it very simply, data transmitted when using OpenVPN is encapsulated and encrypted inside an encrypted tunnel.

The Onion Router

'The Onion Router', or TOR, is a distributed overlay network which aims to anonymize online traffic. TOR consists of a big network of relays, or nodes, through which data will be passed. When connected to the TOR network, the data sent will be layered in multiple layers of encryption and pass through a randomly chosen route of nodes between the source and destination.
The *onion* in the name refers to the layered encryption, which is much like the layers of an onion, with the data contained in the middle.
The encryption TOR uses is based on a stream cipher – 128-bit AES in counter-mode, the Diffie-Hellman protocol, a public key cipher based on 1024-bit RSA, and a hash function: SHA1.

Structure and Path in the TOR network

The first thing that happens when data is sent over TOR is that a random path is chosen, and the data will be layered in the same amount of layers of encryption as the amount of nodes. Each node in the path then decrypts the top layer, until it reaches the final node in the path called the *exit node*, where the last layer of encryption is decrypted, and it then reaches it destination.
Each node in the path only knows the previous node from which the data came, and which node is the next one in its path, but no node knows the whole path.

Weaknesses and Ethics

Because all of these methods make for a very complicated way of decryption – SSL requiring keys for decryption, and TOR being layered in several layers of encryption – trying to decrypt them is resource-wise not a good option, and thus one has to focus on their weaknesses.

TOR has a rather obvious weakness: as each node decrypts one layer, the final node decrypts the final layer before it reaches its destination, which means that unless an end-to-end encryption such as TLS or SSL/TLS is used, traffic will be unencrypted at the final step. As such, wiretapping the *exit node* may be the only feasible way to monitor traffic passing through the TOR network.

SSL has one of its main weaknesses in the user – because if a certificate with mismatching keys is presented from the server, most browsers will warn the user about this, but this is commonly disregarded by the user since many servers use outdated certificates, which can result in the same kind of warning, and thus the chance that the user will accept a mismatching certificate is fairly high.

This means that if a third party would act as a kind of invisible proxy in between the server and the client, and provide the user with a false certificate pretending to be the server, the user may disregard this as an outdated certificate and accept it regardless – ultimately accepting the third party as a relay between the user and the server.

However, SSL is made to be resistant to brute-force attacks (although one should note that it's not impossible, successful attempts have been made), and to make it very hard to decrypt without access to the proper keys. TOR, on the other hand, is resistant to brute force attacks because of its multiple layers of encryption and its implementation of several different components to apply it. Both SSL and TOR have weaknesses that lie outside the scope of what can be done without directly attacking either client, or server. Although TOR exit nodes located within Europe may prove to be a vulnerability to the TOR-network in the future, since data leaving the exit nodes may be logged and stored according to the Data Retention Directive, as of now the weakness of TOR lies in whether or not the operator of the exit nodes are trusted.

Although all of these protocols are useful to encrypt and ensure safer communication between those who wish to do so, they say any tool can become a weapon in the wrong hands. But does that mean that any pair of hands a tool may end up should be assumed to be the wrong ones, and is it really ethical to wiretap TOR exit nodes, or fake SSL certificates and trick users, in order to make sure nobody slips through the cracks of these laws, or is it perhaps the next step in making sure the laws that are being proposed and/or introduced can be enforced as they were intended to be?

Do we have to redefine our ethics in favor of these laws, or have these laws been ethically defined – and if so, does it justify use of these techniques to enforce them?