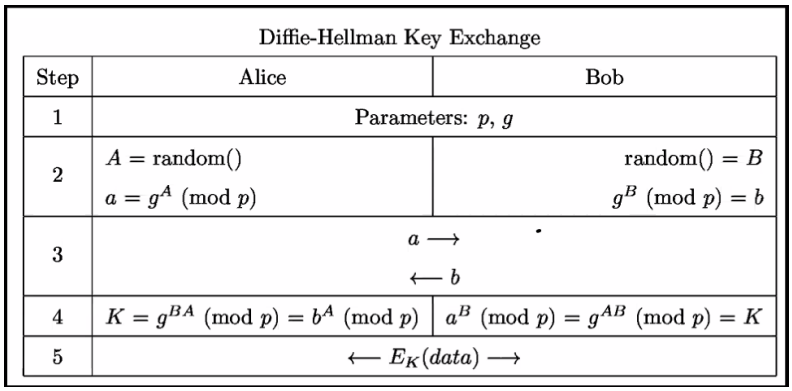
Ricky Marin Network Security Part II TOR April 14, 2020

Building a Mini Tor Network

- TLS 1.3 Session Setup
    1. Client Server interact with Diffie-Hellman Key Exchange
    2. Use libraries, never implement yourself.
- Diffie-Hellman Key Exchange
    1. Alice and Bob agree beforehand on common paint, but each have their own secret colors. They mix with common paint. When they share that with each other, and mix their secret color with the other persons mixture, they arrive at the common secret.
    2. Alice and Bob generate their own random numbers, their "secret colors". They raise their number to g, which was the common "paint". They transmit to each other. They then raise that to their own random number, which gives the key. There is a mod(p) to each step, where p is another common "paint"
    3. You then put the key in a hash function with two nonces to avoid replicability attacks
    4. 

Diffie-Hellman Key Exchange

Step	Alice	Bob
1	Parameters: p, g	
2	$A = \text{random}()$ $a = g^A \pmod p$	$\text{random}() = B$ $g^B \pmod p = b$
3	$a \longrightarrow$ $\longleftarrow b$	
4	$K = g^{BA} \pmod p = b^A \pmod p$	$a^B \pmod p = g^{AB} \pmod p = K$
5	$\longleftarrow E_K(data) \longrightarrow$	

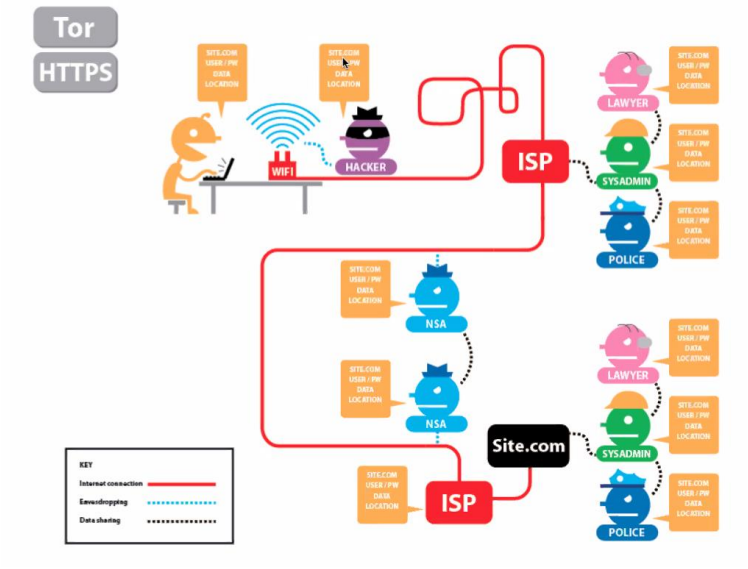
    5. You can not just take the log_g(a) because the g is very very big. For parameters actually used in industry, this is 2^128 powers.
- The server comes back with a nonce, KeyShare and an Encoded Certificate
- Steps.
    1. Get public key from Facebook
    2. Create Temporary private key public key pair
    3. Calculate temporary symmetric key K= f(g^(BA))
    4. Gives public key and encrypted message.
    5. Facebook calculates temporary symmetric key f(g^(AB))
    6. Facebook decrypts message using temporary symmetric key.
    7. Browser can verify message by decrypting
- Writing a secure socket client

1.
```python
import socket, ssl

HOST, PORT = '127.0.0.1', 443

def handle(conn):
    conn.write(b'GET / HTTP/1.1\n')
    print(conn.recv().decode())

def main():
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    context = ssl.create_default_context(ssl.Purpose.SERVER_AUTH)
    context.options |= ssl.OP_NO_TLSv1 | ssl.OP_NO_TLSv1_1  # optional
    conn = context.wrap_socket(sock, server_hostname=HOST)
    try:
        conn.connect((HOST, PORT))
        handle(conn)
    finally:
        conn.close()

if __name__ == '__main__':
    main()
```

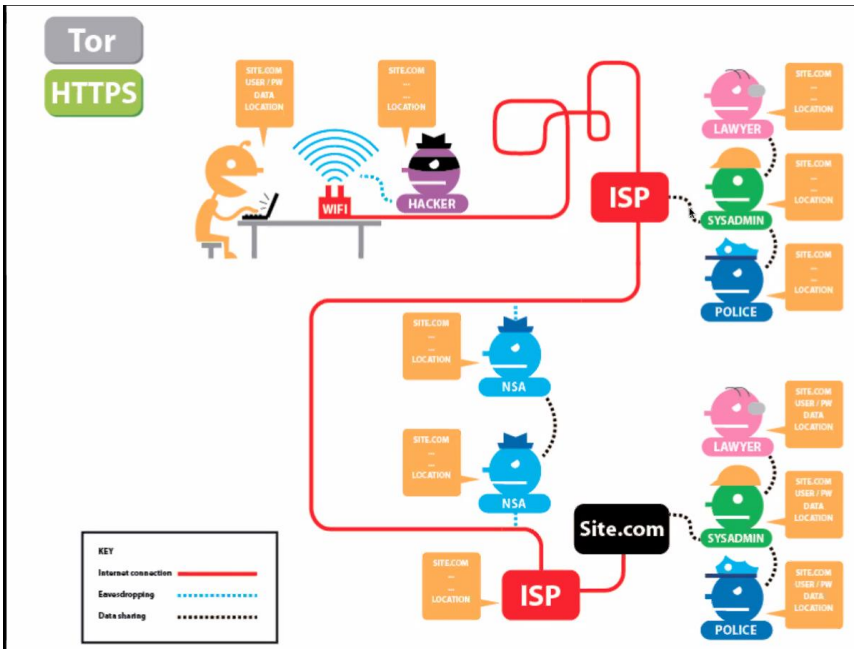
- How to generate keys with OpenSSL
    1.
```
openssl req -new -x509 -days 365 -nodes -out cert.pem -keyout key.pem
```
- Writing a secure socket server
    1.


- Mini Tor
    1. Entry Guard, Middle Relay, Middle Relay, and Exit Relay is all encrypted by Tor, while the connection from Exit Relay to Server is not.
    2. Tor Directory manages all entry nodes and sometimes middle nodes.
        - You can then create a TLS connection to one of the entry nodes
        - The entry guard then connects to the middle relay and wraps the connection.
        - “Onions” multiple layers
    3. Tails has the Tor Browser Bundle
        - Good Security OS

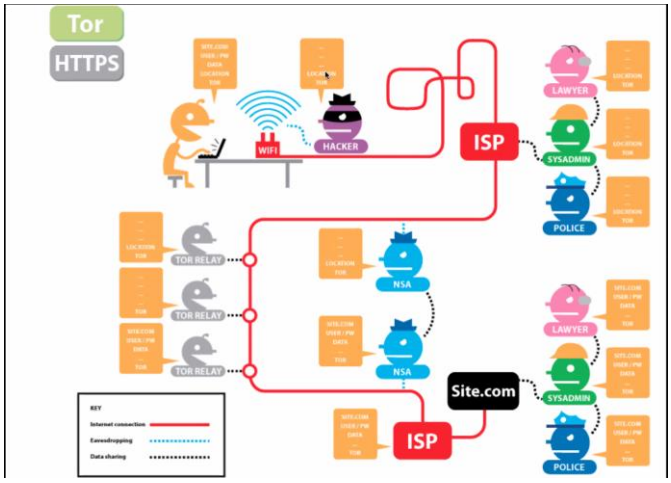
4. What is Hidden, No Tor No HTTPS

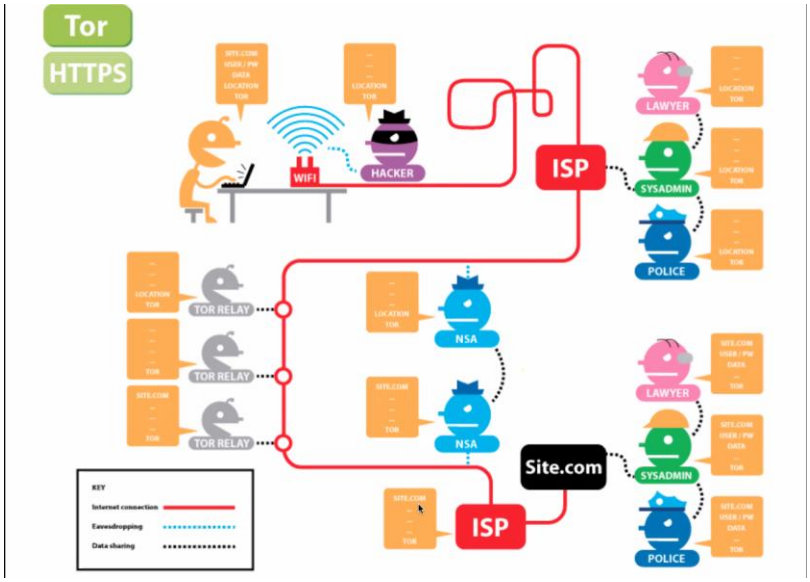


5. Just HTTPS



6. Just Tor, but not HTTPS

- 

7. Both Tor and HTTPS
   - 

     - NSA can still access the websites you were on.
8. Anonabox produces routers that can route all traffic through Tor