THE TLS & TOR PROTOCOLS

Eike Ritter

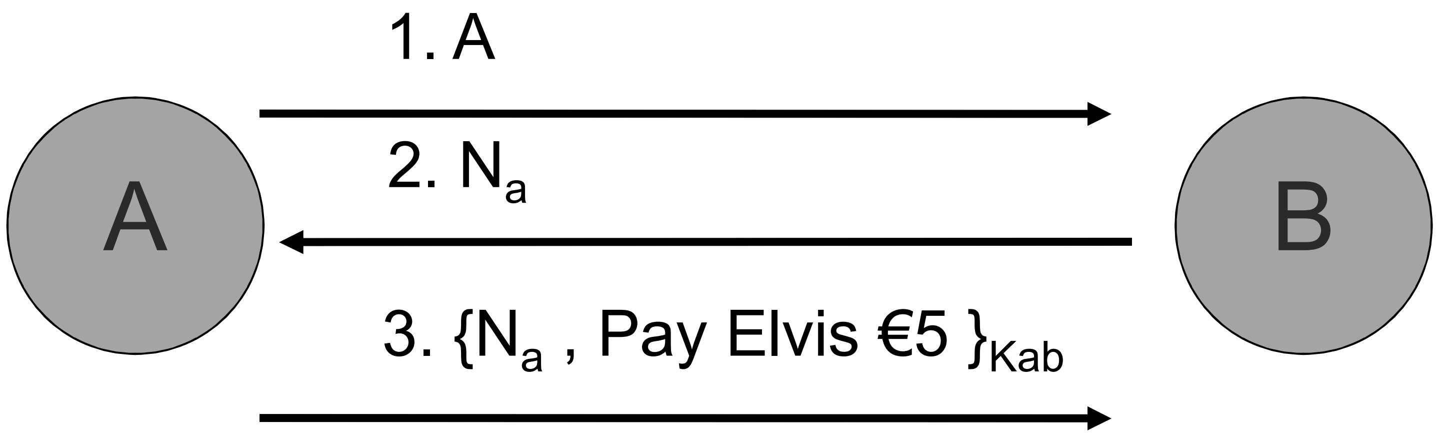
Computer Security and Networks

Introduction

- Details of TLS
  - How it works,
  - Common problems

- Tor,
  - Anonymity on the Internet
  - The Tor Protocol
  - Hidden servers.

Last Lecture



1. $A \rightarrow B : A$
2. $B \rightarrow A : N_a$
3. $A \rightarrow B : \{N_a, \text{Pay Elvis €5}\}_{Kab}$

Last Lecture

Assume Alice and Bob know each others public keys, can they set up a symmetric key?

1. $A \rightarrow B : E_B(N_a, A)$
2. $B \rightarrow A : E_A(N_a, N_b)$
3. $A \rightarrow B : E_B(N_b)$

$E_X(\_)$ means public key encryption

N_a and N_b can then be used to generate a symmetric key

Last Lecture

The attacker acts as a man-in-the-middle:

1. $A \rightarrow C : E_C(N_a, A)$

    1\`. $C(A) \rightarrow B : E_B(N_a, A)$

    2\`. $B \rightarrow C(A) : E_A(N_a, N_b)$

2. $C \rightarrow A : E_A(N_a, N_b)$
3. $A \rightarrow C : E_C(N_b)$

    3\`. $C(A) \rightarrow B : E_B(N_b)$

Last Lecture

A very simple fix:

1. $A \rightarrow B : E_B(N_a, A)$
2. $B \rightarrow A : E_A(N_a, N_b, B)$
3. $A \rightarrow B : E_B(N_b)$

The SSL/TLS Protocol

- The Secure Sockets Layer (SSL) protocol was renamed the Transport Layer Security (TLS) protocol.
- It provides encrypted socket communication and authentication, based on public keys.
- It may use a range of ciphers (RSA,DES,DH,..)
  - These are negotiated at the start of the run.

X.509 Standard for Certificates

X.509 certificates contain a subject, subject’s public key, Issuer name, etc

The issuer signs the hash of all the data

To check a certificate I hash all the data and check the issuers public key.

If I have the issuer’s public key, and trust the issuer, I can then be sure of the subject’s public key.

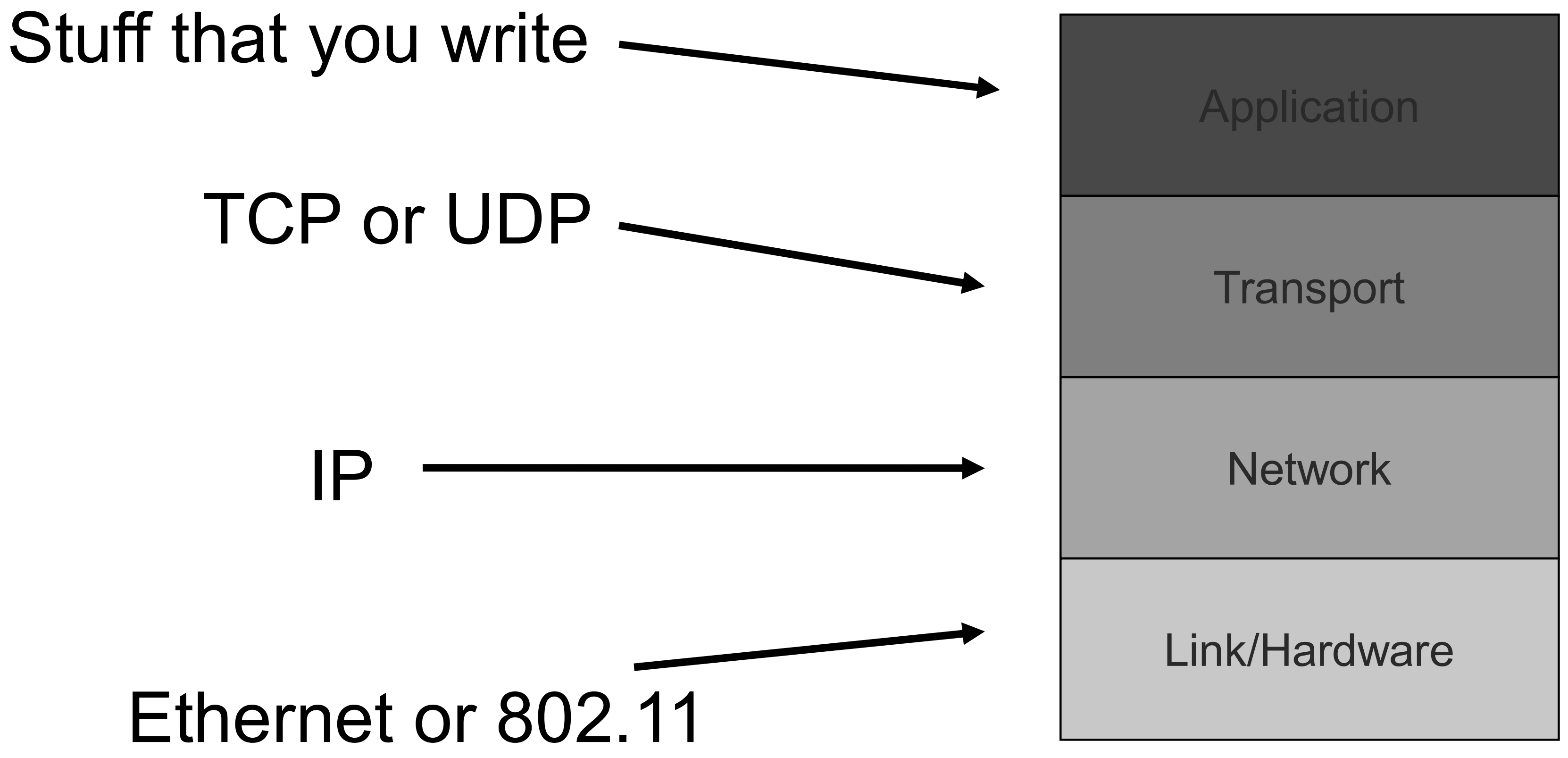
Example Cert.

- In firefox https google and facebook.

- FireFox > Preferences > Advanced > Certificates >

> View Cert.

The Internet Protocol Stack, (Most of the Time):

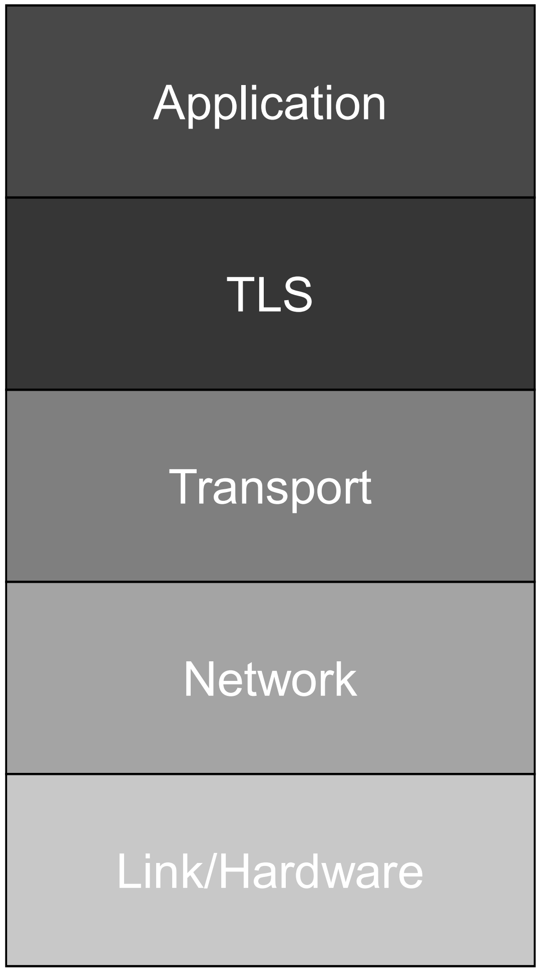


The Internet Protocol Stack with TLS

The TLS layer runs between the Application and Transport layer.

The encryption is transparent to the Application layer.

Normal TCP and IP protocols etc. can be used at the low layers



Transport Layer Security (TLS)

1. $C \rightarrow S : N_C$
2. $S \rightarrow C : N_S$, $Cert_S$
3. $C \rightarrow S : E_S(K\_seed), \{Hash1\}_{K_{CS}}$
4. $S \rightarrow C : \{Hash2\}_{K_{CS}}$

Hash 1 = $\#(N_C, N_S, E_S(K\_seed))$

Hash 2 = $\#(N_C, N_S, E_S(K\_seed), \{Hash1\}_{K_{CS}})$

K_{CS} is a session key based on N_C, N_S & K_seed,
All data is then encrypted with K_{CS} and hashed for integrity.

TLS-DHE

A variant uses Diffie-Hellman for “forward secrecy”

i.e., if someone gets the servers key later they can’t go back and break a recording of the traffic.

1. $C \rightarrow S : N_C$
2. $S \rightarrow C : N_S$, g^x, $Cert_S$, $Sign_S(\#(N_C,N_S,g^x))$
3. $C \rightarrow S : g^y$, $\{\#(\text{All previous messages})\}_{K_{CS}}$
4. $S \rightarrow C : \{\#(\text{All previous messages})\}_{K_{CS}}$

K_{CS} is a session key based on N_C, N_S & g^{xy},

Cipher Suites

- Cipher Suites with encryptions and authentication:

```
SSL_RSA_WITH_3DES_EDE_CBC_SHA
SSL_RSA_WITH_DES_CBC_SHA
SSL_RSA_WITH_RC4_128_MD5
SSL_RSA_WITH_RC4_128_SHA
TLS_DHE_DSS_WITH_AES_128_CBC_SHA
TLS_DHE_DSS_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA
TLS_KRB5_EXPORT_WITH_DES_CBC_40_MD5
TLS_KRB5_EXPORT_WITH_DES_CBC_40_SHA
...
```

- Cipher Suites with just authentication:

```
SSL_RSA_WITH_NULL_MD5
SSL_RSA_WITH_NULL_SHA
```

- Cipher Suites with just encryptions:

```
SSL_DH_anon_EXPORT_WITH_DES40_CBC_SHA
SSL_DH_anon_EXPORT_WITH_RC4_40_MD5
SSL_DH_anon_WITH_3DES_EDE_CBC_SHA
SSL_DH_anon_WITH_DES_CBC_SHA
SSL_DH_anon_WITH_RC4_128_MD5
TLS_DH_anon_WITH_AES_128_CBC_SHA
TLS_DH_anon_WITH_AES_256_CBC_SHA
```

TLS demo

- Websites
- Wireshark

Weaknesses in TLS

- Configuration weaknesses:
  - Cipher downgrading
  - Self signed certificates

- Direct attack against implementations:
  - Apple’s goto fail bug.
  - LogJam attack
  - HeartBleed

Cipher downgrading attack

- Client supports:

```
TLS_DHE_DSS_WITH_AES_256_CBC_SHA
TLS_DHE_DSS_WITH_AES_128_CBC_SHA
SSL_RSA_WITH_3DES_EDE_CBC_SHA
SSL_RSA_WITH_DES_CBC_SHA
```

- Server supports:

```
TLS_DHE_DSS_WITH_AES_128_CBC_SHA
SSL_RSA_WITH_3DES_EDE_CBC_SHA
SSL_RSA_WITH_DES_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA
TLS_KRB5_EXPORT_WITH_DES_CBC_40_MD5
TLS_KRB5_EXPORT_WITH_DES_CBC_40_SHA
```

What cipher will be used?

BUT…

Cipher downgrading attack

- Client supports:

~~TLS_DHE_DSS_WITH_AES_256_CBC_SHA~~

~~TLS_DHE_DSS_WITH_AES_128_CBC_SHA~~

~~SSL_RSA_WITH_3DES_EDE_CBC_SHA~~

$\boxed{\texttt{SSL\_RSA\_WITH\_DES\_CBC\_SHA}}$

The cipher suite messages
are not authenticated

- Server supports:

~~TLS_DHE_DSS_WITH_AES_128_CBC_SHA~~

~~SSL_RSA_WITH_3DES_EDE_CBC_SHA~~

$\boxed{\texttt{SSL\_RSA\_WITH\_DES\_CBC\_SHA}}$

TLS_KRB5_EXPORT_WITH_DES_CBC_40_MD5

TLS_KRB5_EXPORT_WITH_DES_CBC_40_SHA

An attacker that owns the network can remove the strong ciphers

If both client and server support a weak cipher
then an attack can force its use.

Self signed certificates

- Maintaining a set of certificates is hard (especially on apps and IoT devices).
- It’s much easier just to accept any certificate, (or certificates that sign themselves).
- What’s the problem?

Self signed certificates

- Maintaining a set of certificates is hard (especially on apps and IoT devices).
- It’s much easier just to accept any certificate, (or certificates that sign themselves).
- If the client accepts the self signed certificates then it’s easy to man-in-the-middle.
- This has been shown to happen a lot in devices and code that uses TLS!

Apple’s Implementation of TLS

http://opensource.apple.com/source/Security/Security-55471/libsecurity_ssl/lib/sslKeyExchange.c

Apple’s TLS-DHE

1. $C \rightarrow S : N_C$
2. $S \rightarrow C : N_S$, g^x, $Cert_S$, $SignS(\#(N_C,N_S,g^x))$
3. $C \rightarrow S : g^y$, $\{\#(\text{All previous messages})\}_{K_{CS}}$
4. $S \rightarrow C : \{\#(\text{All previous messages})\}_{K_{CS}}$

K_{CS} is a session key based on N_C, N_S & g^{xy},

Apple’s TLS-DHE

1. $C \rightarrow S : N_C$
2. $S \rightarrow C : N_S , g^x, Cert_S , True$
3. $C \rightarrow S : g^y, \{\#(\text{All previous messages})\}_{K_{CS}}$
4. $S \rightarrow C : \{\#(\text{All previous messages})\}_{K_{CS}}$

K_{CS} is a session key based on N_C, N_S & g^{xy},

Major issues

- iOS fixed days before Mac OS!
- Why didn't tests pick this up?
- Compiler should warned of unreachable code.
- Bad programming style: no brackets, goto:

I COULD RESTRUCTURE THE PROGRAM'S FLOW
OR USE ONE LITTLE 'GOTO' INSTEAD.
EH, SCREW GOOD PRACTICE. HOW BAD CAN IT BE?
goto main_sub3;
COMPILE

http://xkcd.com/292/

Cipher Suites

- What if one side supports a weak cipher suite but the other does not?

Cipher Suites

- What if one side supports a weak cipher suite but the other does not?
- Generally considered safe.
- Browser developers removed all weak ciphers, some remained in servers.
- This depends on different cipher suites being incompatible, e.g.:
- **`SSL_RSA_WITH_DES_CBC_SHA`** and **`TLS_DHE_DSS_WITH_AES_256_CBC_SHA`**

LogJam

- The Snowdon leaks revealed that the NSA regularly MITMed TLS.
- How could they be doing this? Someone had missed something.
- This people figured it out:

  https://weakdh.org/imperfect-forward-secrecy-ccs15.pdf
- A weak Diffie Hellman key is compatible with a strong Diffie Hellman key!

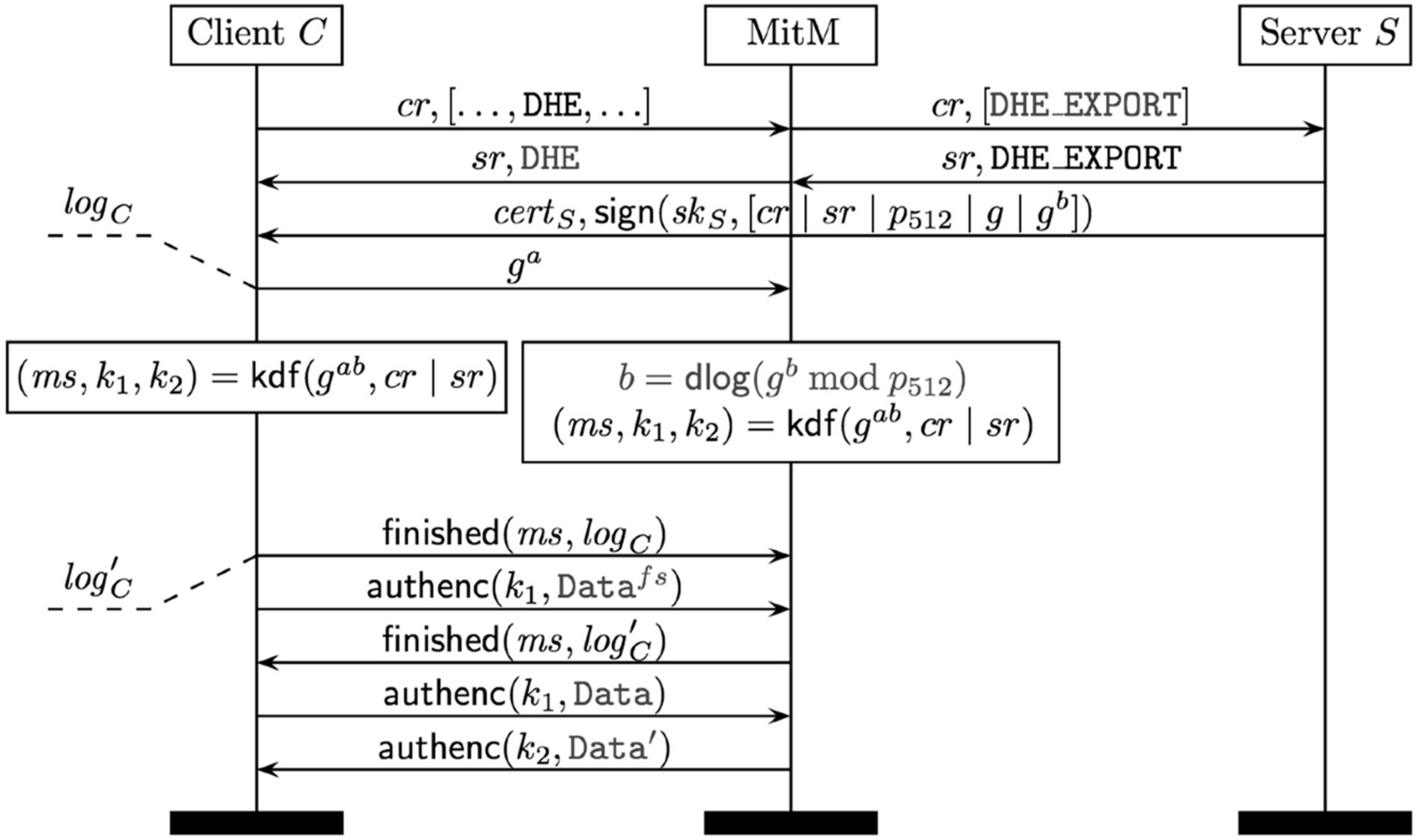
Diffie-Hellman

- Alice and Bob pick random numbers r_A and r_B and find "$t_A = g^{r_A} \bmod p$" and "$t_B = g^{r_B} \bmod p$"

- The protocol just exchanges these numbers:
  1. $A \rightarrow B : t_A$
  2. $B \rightarrow A : t_B$

- "Alice" calculates "$t_B^{r_A} \bmod p$" and "Bob" "$t_A^{r_B} \bmod p$" this is the key:
  - $K = g^{r_A r_B} \bmod p$

Attack Diagram from paper above



Client C
MitM
Server S
$cr, [\ldots, \mathtt{DHE}, \ldots]$
$cr, [\mathtt{DHE\_EXPORT}]$
$sr, \mathtt{DHE}$
$sr, \mathtt{DHE\_EXPORT}$
log_C
$cert_S, \mathsf{sign}(sk_S, [cr \mid sr \mid p_{512} \mid g \mid g^b])$
g^a
$(ms, k_1, k_2) = \mathsf{kdf}(g^{ab}, cr \mid sr)$
$b = \mathsf{dlog}(g^b \bmod p_{512})$
$(ms, k_1, k_2) = \mathsf{kdf}(g^{ab}, cr \mid sr)$
$\mathsf{finished}(ms, log_C)$
log'_C
$\mathsf{authenc}(k_1, \mathtt{Data}^{fs})$
$\mathsf{finished}(ms, log'_C)$
$\mathsf{authenc}(k_1, \mathtt{Data})$
$\mathsf{authenc}(k_2, \mathtt{Data}')$

HeartBleed

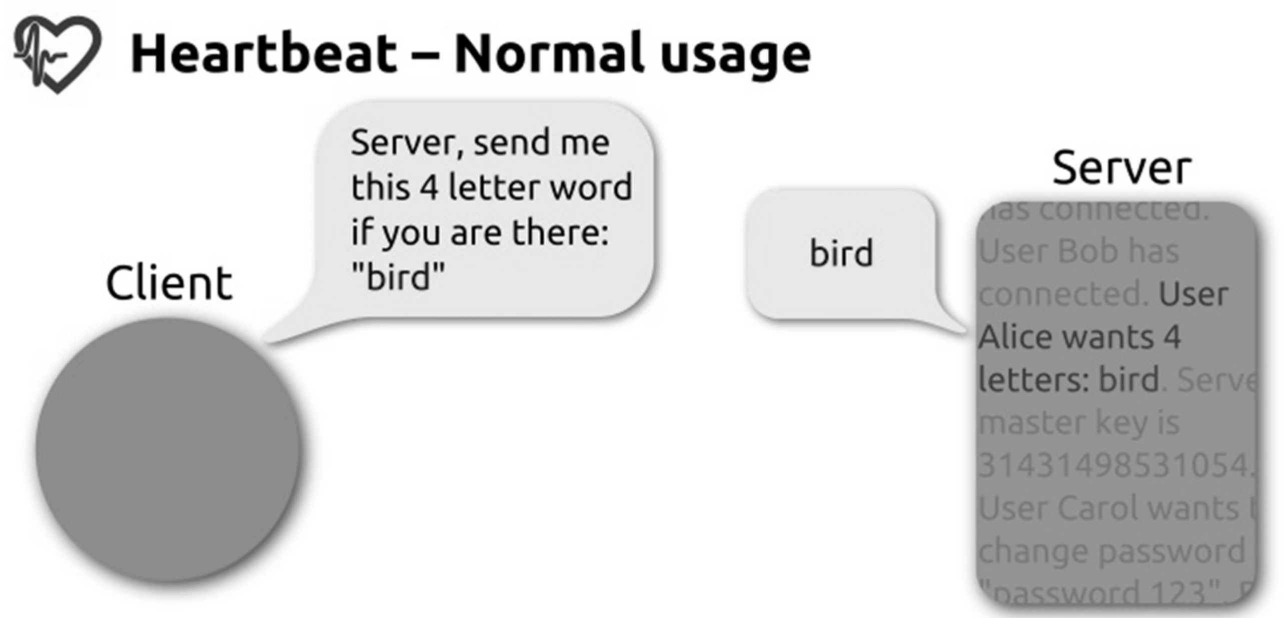
A programing error in OpenSSL

Introduced in 2012, made public in 2014.
Rumors it was being exploited.

TLS client can request a “heart beat” from
The server to make sure the connection is still open.

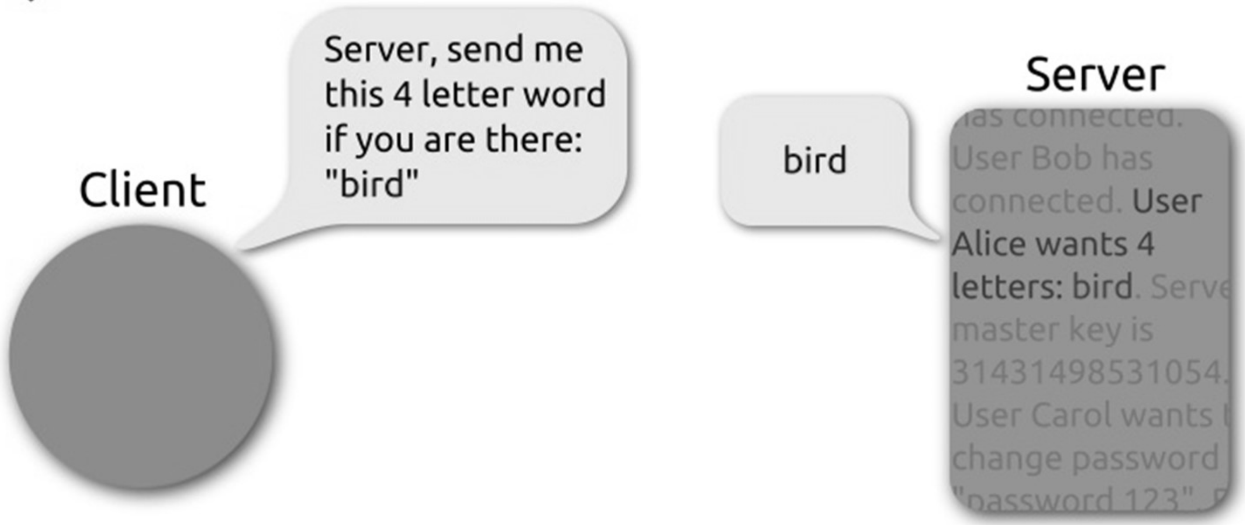
The client could ask for a much longer message and would be given parts of the servers memory.

This memory could contain the servers key

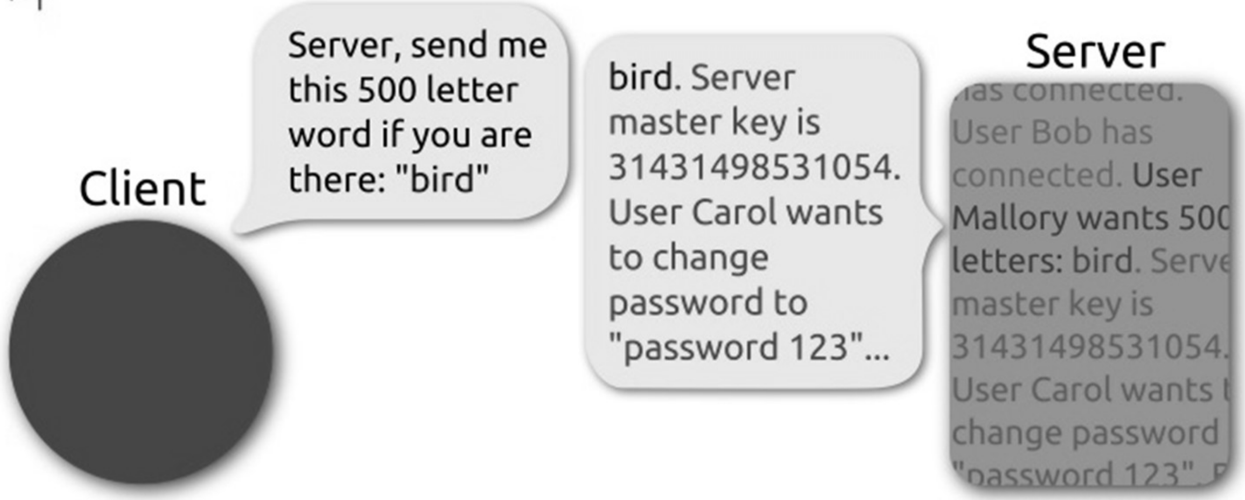

Heartbeat – Normal usage
Client
Server, send me this 4 letter word if you are there: "bird"
Server
bird
has connected. User Bob has connected. User Alice wants 4 letters: bird. Serve master key is 31431498531054. User Carol wants change password "password 123".

From Wikipedia

Heartbeat – Normal usage


Client
Server, send me this 4 letter word if you are there: "bird"
bird
Server
has connected. User Bob has connected. User Alice wants 4 letters: bird. Serve master key is 31431498531054. User Carol wants t change password "password 123". P

Heartbeat – Malicious usage

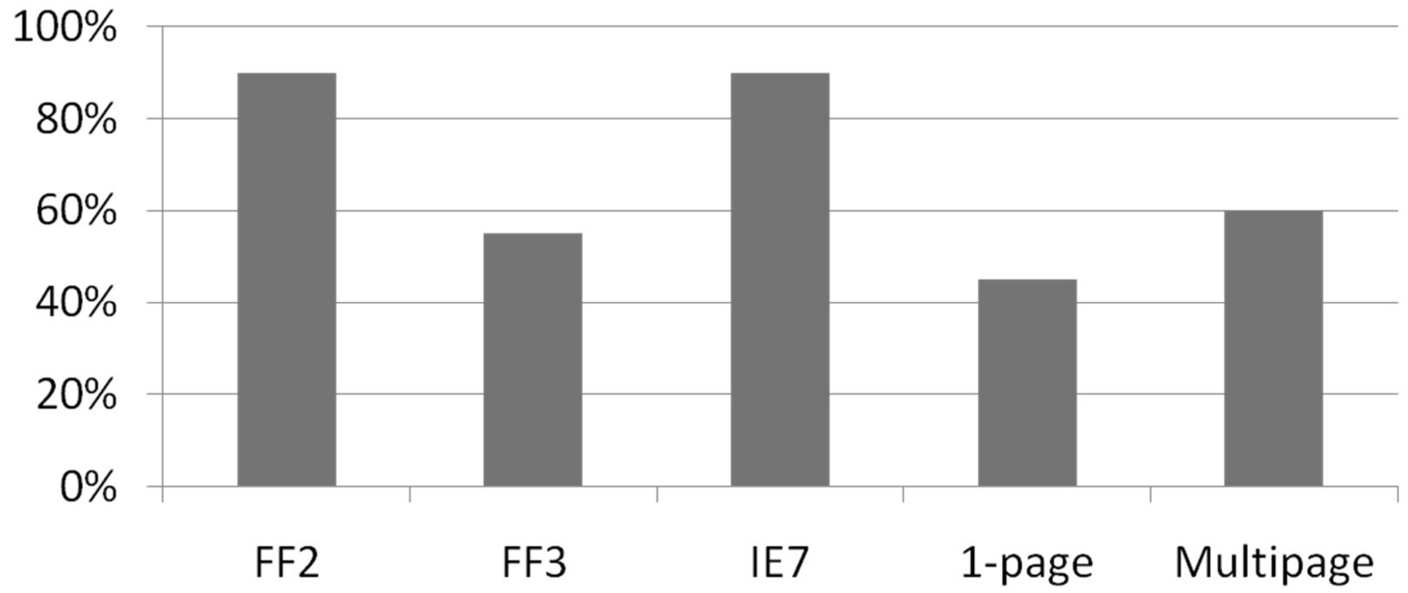

Client
Server, send me this 500 letter word if you are there: "bird"
bird. Server master key is 31431498531054. User Carol wants to change password to "password 123"...
Server
has connected. User Bob has connected. User Mallory wants 500 letters: bird. Serve master key is 31431498531054. User Carol wants t change password "password 123". P

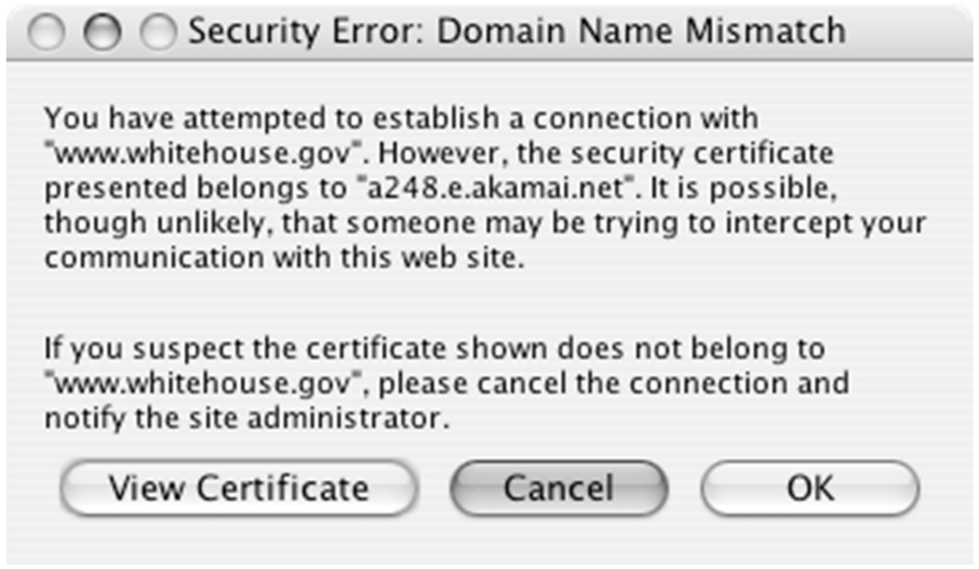
TLS 1.3

- Newest standard, ratified August 2018
- Removes obsolete cryptographic protocols
- Simplified handshake => efficiency gain
- Forward secrecy mandatory
- Intercepting TLS connections now only possible as active attacker performing MITM attack

Cranor et al.'s Crying Wolf: "An Empirical Study of SSL Warning Effectiveness."

People That Ignored Warnings

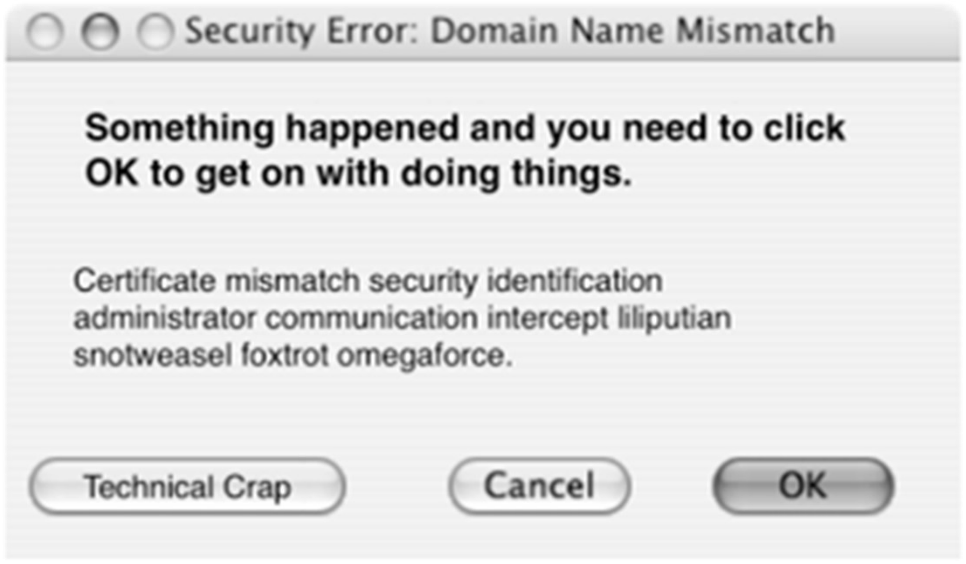



Security Error: Domain Name Mismatch

You have attempted to establish a connection with "www.whitehouse.gov". However, the security certificate presented belongs to "a248.e.akamai.net". It is possible, though unlikely, that someone may be trying to intercept your communication with this web site.

If you suspect the certificate shown does not belong to "www.whitehouse.gov", please cancel the connection and notify the site administrator.

Image courtesy of Johnathan Nightingale


Security Error: Domain Name Mismatch

Something happened and you need to click OK to get on with doing things.

Certificate mismatch security identification administrator communication intercept liliputian snotweasel foxtrot omegaforce.

Image courtesy of Johnathan Nightingale

Checking servers

- There are many insecure TLS servers on the Internet.

- The most common problems are support for weak ciphers and old unpatched code.

- SSL labs provide a useful testing tool:
  - https://www.ssllabs.com/ssltest/index.html

Introduction

- Details of TLS
  - How it works,
  - Common problems

- Tor,
  - Anonymity on the Internet
  - The Tor Protocol
  - Hidden servers.

“YOU HAVE ZERO PRIVACY ANYWAY, GET OVER IT”

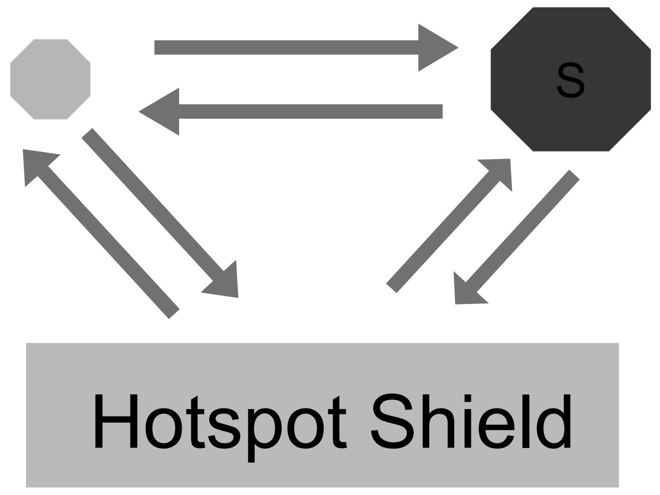
Scott McNealy,

CEO of SUN Microsystems.

“With your permission, you give us more information about you, about your friends, and we can improve the quality of our searches. We don't need you to type at all. We know where you are. We know where you've been. We can more or less know what you're thinking about.”

- Eric Schmidt, CEO of Google

Proxy: Hotspot Shield VPN



An Internet connection reveals your IP number.

The promises “Anonymity”

Connection made via their servers.

The server never see’s your IP address.

Virtual Private Networks

- VPNs securely connect you to another network.
- E.g. you can connect to the schools printers via the schools VPN
- Secured with certificates and encryption, e.g. TLS, or IPsec.

Virtual Private Networks For Anonymity

- To get some anonymity you can route all your traffic via the VPN.
  - Server thinks you are the VPN provider
  - ISP only sees the connection to the VPN
  - A global observer can probably link your connections.
- There is no anonymity to the VPN.
- E.g. you can connect to the the schools prints via the schools VPN
- Secure with Certificates and encryption, e.g. TLS, or IPsec.

Onion Routing

- You get the best anonymity by routing your traffic via a number of proxies.
- Onion Routing ensures that your message really is routed via the proxies you want.
- The TOR network is using this protocol
  https://www.torproject.org

TOR: Onion Routing

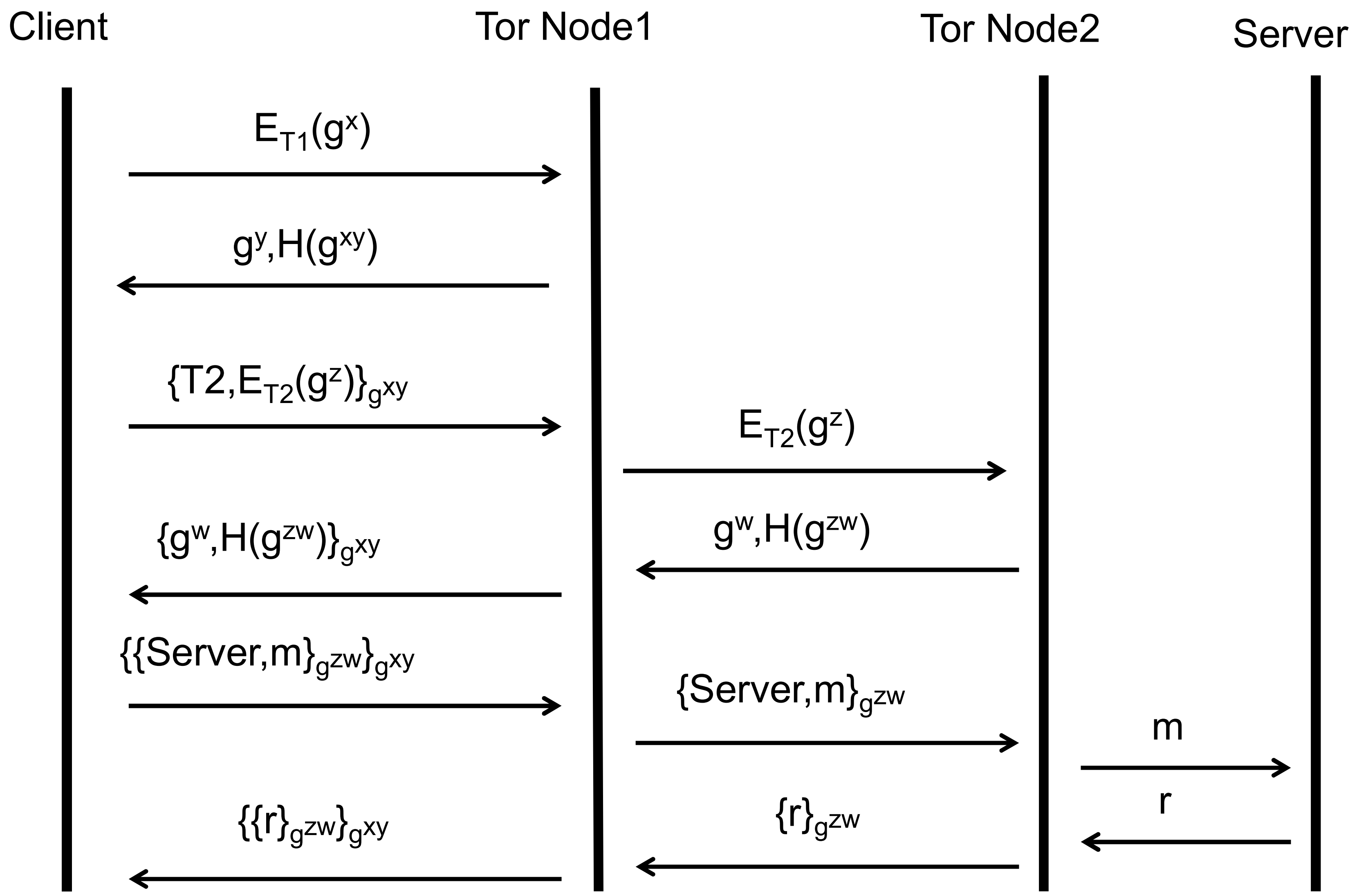
- Each proxy only learns the IP of the proxy before it and the proxy after it.

- The public key of each proxy is known.

- Source IP is visible to the first node, destination IP is visible to the last node.

- User picks 3 proxies, and is anonymous as they aren’t all corrupt.

Tor Onion Routing



Hidden Servers

- Tor hidden servers hides the server from you.
- See: https://www.torproject.org/docs/hidden-services.html.en

Anonymity does not equal security

- The webservers can still be attacked.
  - E.g. FBI attack attacked criminal websites running on Tor and implanted malware.

- Poor use of Tor also
  - http://arstechnica.com/security/2013/12/use-of-tor-helped-fbi-finger-bomb-hoax-suspect/
  - See warning on: https://securedrop.theguardian.com

Conclusion

- Details of TLS
  - How it works,
  - Common problems

- Tor,
  - Anonymity on the Internet
  - The Tor Protocol
  - Hidden servers.