A Cross-Protocol Attack on the TLS Protocol

Nikos Mavrogiannopoulos, Frederik Vercauteren , VesselinVelichkov, Bart Preneel.
ESAT/SCD/COSIC – IBBT

Presentation by Lin Wang
USF
lwang20@usfca.edu

TLS protocol SAFE or Not??

Overview

- TLS protocol
- The Wagner and Schneier Attack.
- A New Cross-Protocol Attack
- Attack Assumptions
- Feasibility of the Attack
- Possible Fix

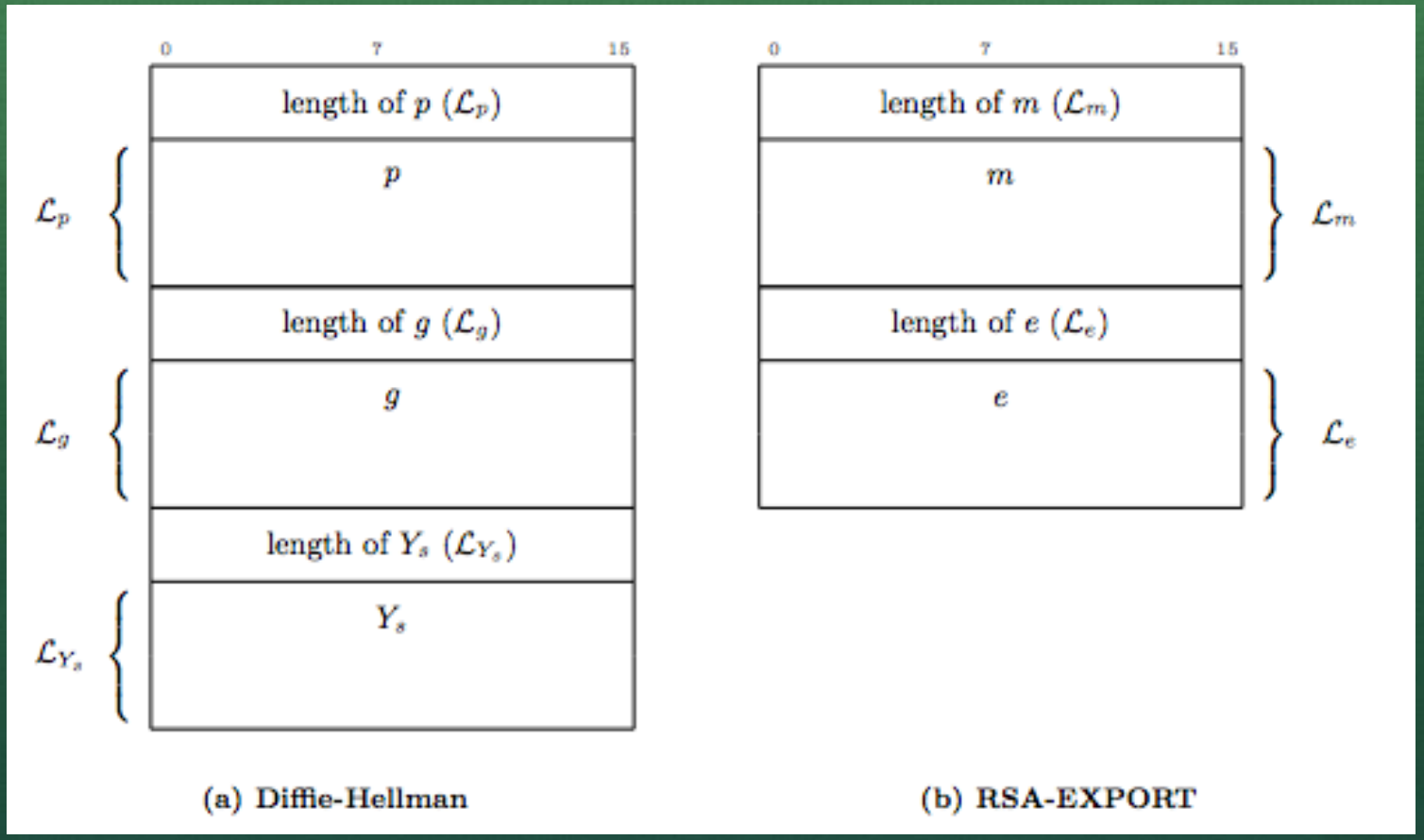
TLS protocol

- Transport Layer Security protocol
- TLS is one of the major secure communications protocols on the Internet, used by a variety of applications such as web browsers, electronic mail, voice over-IP and more.
- Ciphersuite: determines the symmetric encryption cipher with its operational mode, the key exchange method and the message authentication algorithm.

The Wagner and Schneier Attack

- Wagner and Schneier attack is a cross-protocol attack based on the observation that the digital signature in a DH key exchange does not cover any identifier of the negotiated ciphersuite.
- The attack deceives a client who advertises a 'TLS – RSA EXPORT' ciphersuite and expects temporary RSA parameters in the 'ServerKeyExchange' message, into receiving DH parameters from a 'TLS DHE RSA' ciphersuite.

contents of the ServerKeyExchange message

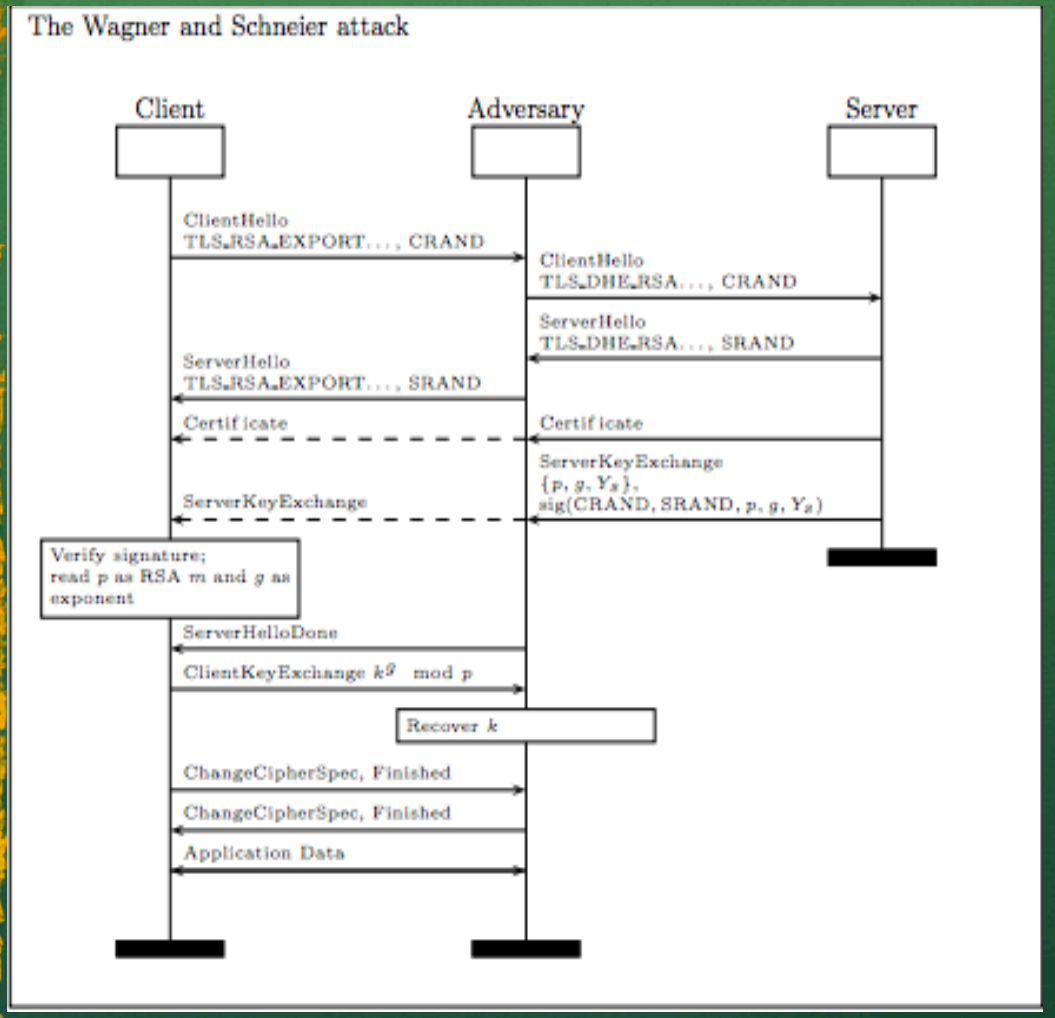


Process of Wagner and Schneier attack

1.Client verifies the signature, reads the RSA modulus m, which corresponds to the prime of the DH group p, and then reads the RSA exponent e field which corresponds to the group generator g.

2.Client encrypts the pre-master secret k as k^g mod p and includes it in its 'ClientKeyExchange' message. Since p is a prime number and g is known, it is very easy to compute the gth root of kg to recover k.

- Attack



Meaning of The Wagner and Schneier Attack

- Careful examination of the TLS packet parsing reveals that the failure of the attack is due to the serialized way TLS packets need to be parsed.

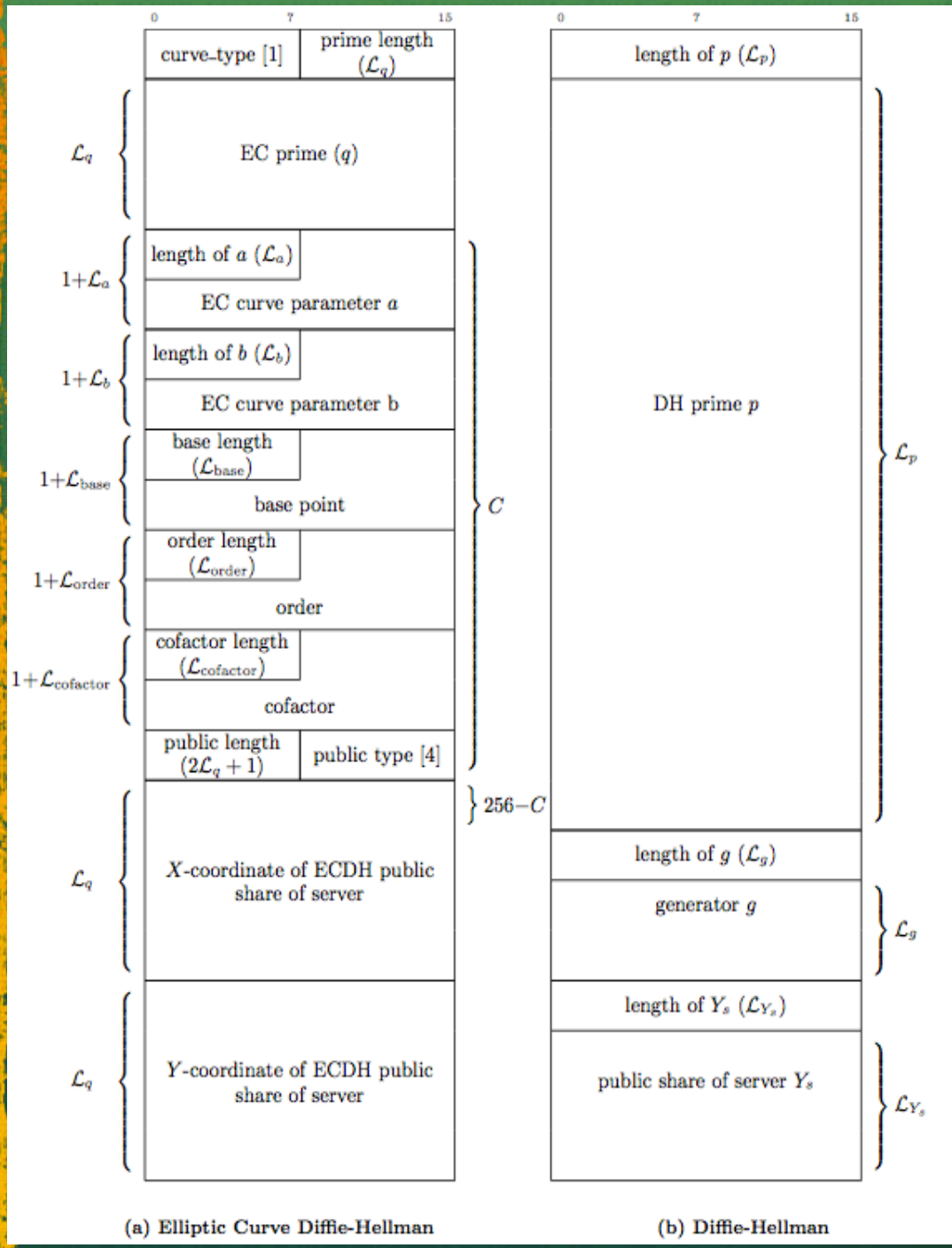
- Even though the Wagner and Schneier attack fails, it demonstrates the idea of a cross-protocol attack utilizing two of the SSL 3.0 key exchange methods, the DH key exchange and the RSA-EXPORT key exchange.

A New Cross-Protocol Attack

- A server impersonation attack on clients that support the DH key exchange and wish to connect to a server that supports, among others, the ECDH method.
- The attack requires the server to support the explicit prime curve option, and the client to support the plain DH method. Because the only common signature algorithm in the ECDH and DH key exchanges is RSA, the server is also required to possess an RSA signing key.

Contrast ServerKeyExchange message

- In the explicit prime curve option, the server includes in its signed 'ServerKeyExchange' message the parameters of its elliptic curve and an ephemeral public key to be used for this session .

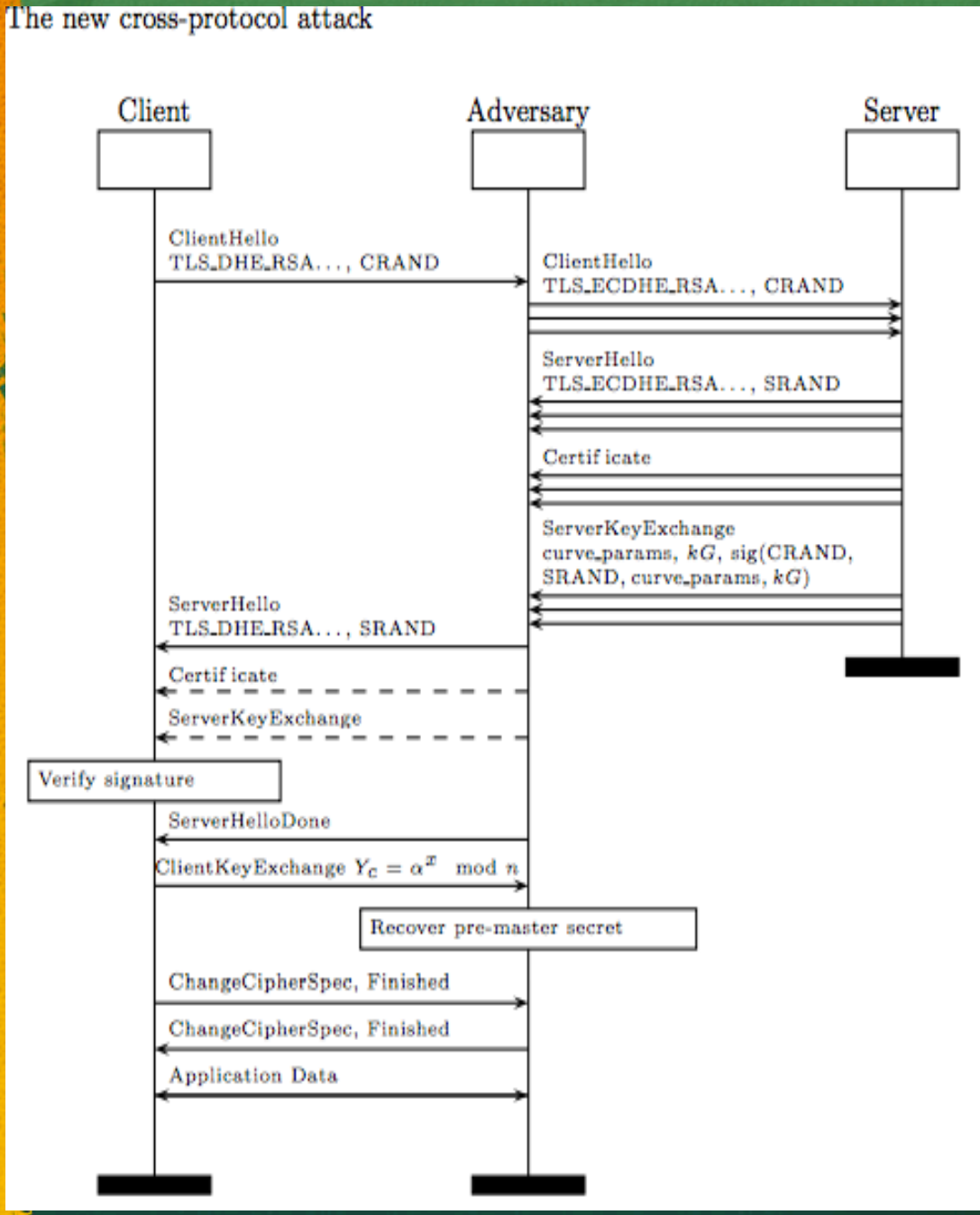


Process of A New Cross-Protocol Attack

ECDH 'ServerKeyExchange' need satisfies two properties.

1.The message can be interpreted as a valid DH 'Server Key Exchange' message.

2.The adversary can recover the exchanged DH key.



A New Cross-Protocol Attack

- **Probability of valid key exchange message**
  - *Length requirements on key exchange parameters.*
  - *Probability estimate.*
- **Recovering the session key**
  - *Computing* x.
  - *Computing pre-master secret.*
- **Attack success probability**
  - **2^40**

Attack Assumptions

- The client software supports one of the ‘TLS DHE - RSA’ ciphersuites and a DH public key (Ys) with value 1 is accepted.
- The server software supports one of the ‘TLS ECDHE - RSA’ ciphersuites, with the ‘arbitrary explicit prime curve’ option, has selected a curve of size between 300 and 400 bits and uses RSA as the signing algorithm.

Feasibility of the Attack

- **Attacking a specific client**
- **Attacking a random client**

Table 4: TLS handshake timeout values in various browsers.

Browser	Handshake timeout
Chrome 20	20 secs
Firefox 10	30 secs
Internet Explorer 8	40 secs
Opera 12	40 secs

Table 5: The resources required by the web server during the attack simulation.

Web server	
Transmitted data	4.7 MB/sec
Received data	1.8 MB/sec
Requests handled	3770 req/sec

Possible Fix

The authors propose to modify the signature of the 'ServerKeyExchange' to include, in addition to explicit identifiers of the algorithms, all the previously exchanged messages.

Our proposed signature for a 'ServerKeyExchange' messageis shown in Fig.6.

```
enum { server (0), client (1) } ConnectionEnd;

enum { dhe_dss (0), dhe_rsa (1),
       ec_diffie_hellman (2)
     } KeyExchangeAlgorithm;

struct {
  select (KeyExchangeAlgorithm) {
    case dhe_dss:
    case dhe_rsa:
      ServerDHParams params;
    case ec_diffie_hellman:
      ServerECDHParams params;
  }
} Parameters;

struct {
  Parameters params;
  digitally-signed struct {
    ConnectionEnd entity;
    opaque handshake_messages<1..2^24-1>;
    KeyExchangeAlgorithm kx_algorithm;
    Parameters params;
  }
} ServerKeyExchange;
```

Figure 6: The proposed format for the ServerKeyExchange message signature. Note that we follow the TLS protocol message description. In particular, the type opaque is used to indicate bytes containing uninterpreted data and arrays of variable length, specified with the <floor..ceiling> notation, are preceded by a number of bytes containing the length of the array.

Q&A

- Any questions?