ON THE SECURITY OF TLS RENEGOTIATION

Douglas Stebila

Joint work with Florian Giesen & Florian Kohlar, Ruhr-Universität Bochum

2012/11/02
QUT

Queensland University of Technology

European Network of Excellence in Cryptology II (ECRYPT II)

Australian Technology Network–German Academic Exchange Service (ATN-DAAD) Joint Research Co-operation Scheme

ON THE SECURITY OF TLS RENEGOTIATION

1. What is TLS?
   Is TLS secure?

2. What is TLS renegotiation?
   Attacks on TLS renegotiation

3. Modelling security of renegotiation
   Analysing security of TLS renegotiation fixes

TLS

WHAT IS TLS?

- Depends on who you ask
- Users:
  - TLS? What's that?
  - SSL? Huh?
  - HTTPS? That's the lock icon, right?
- Cryptographers:
  - "TLS is perhaps the Internet's most widely used security protocol"
  - 'A key exchange and encryption protocol'
  - 'RSA key transport or signed Diffie–Hellman combined with (authenticated) encryption'

WHAT IS TLS?

> The Transport Layer Security protocol provides communications privacy over the Internet. The protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery.

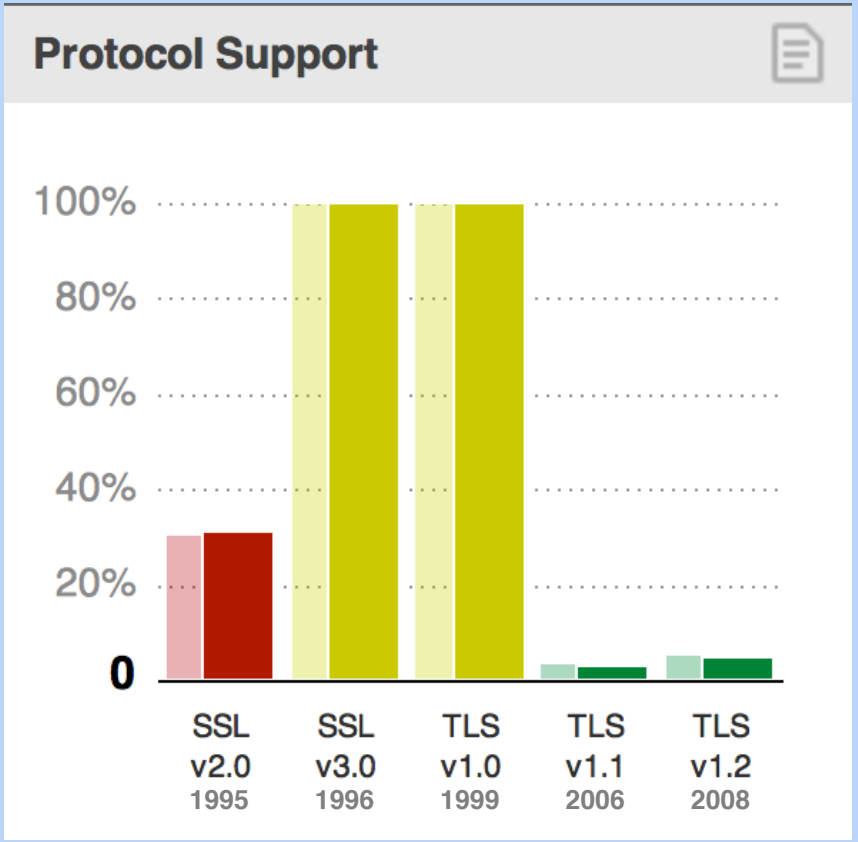
The TLS Protocol Version 1.0

RFC 2246

WHAT IS TLS?

In reality:

- 5 protocol versions
- vast array of standards
- many implementations!
- 300+ combinations of cryptographic primitives
- different levels of security
- different modes of authentication
- additional functionality:
  - alerts & errors
  - session resumption
  - renegotiation
  - compression



https://www.trustworthyinternet.org/ssl-pulse/
August 10, 2012

The current approved version of TLS is version 1.2, which is specified in:

- RFC 5246: “The Transport Layer Security (TLS) Protocol Version 1.2”.

The current standard replaces these former versions, which are now considered obsolete:

- RFC 2246: “The TLS Protocol Version 1.0”.
- RFC 4346: “The Transport Layer Security (TLS) Protocol Version 1.1”.

as well as the never standardized SSL 3.0:

- RFC 6101: “The Secure Sockets Layer (SSL) Protocol Version 3.0”.

Other RFCs subsequently extended TLS.

Extensions to TLS 1.0 include:

- RFC 2595: “Using TLS with IMAP, POP3 and ACAP”. Specifies an extension to the IMAP, POP3 and ACAP services that allow the server and client to use transport-layer security to provide private, authenticated communication over the Internet.
- RFC 2712: “Addition of Kerberos Cipher Suites to Transport Layer Security (TLS)”. The 40-bit cipher suites defined in this memo appear only for the purpose of documenting the fact that those cipher suite codes have already been assigned.
- RFC 2817: “Upgrading to TLS Within HTTP/1.1”, explains how to use the Upgrade mechanism in HTTP/1.1 to initiate Transport Layer Security (TLS) over an existing TCP connection. This allows unsecured and secured HTTP traffic to share the same well known port (in this case, http: at 80 rather than https: at 443).
- RFC 2818: “HTTP Over TLS”, distinguishes secured traffic from insecure traffic by the use of a different 'server port'.
- RFC 3207: “SMTP Service Extension for Secure SMTP over Transport Layer Security”. Specifies an extension to the SMTP service that allows an SMTP server and client to use transport-layer security to provide private, authenticated communication over the Internet.
- RFC 3268: “AES Ciphersuites for TLS”. Adds Advanced Encryption Standard (AES) cipher suites to the previously existing symmetric ciphers.
- RFC 3546: “Transport Layer Security (TLS) Extensions”, adds a mechanism for negotiating protocol extensions during session initialisation and defines some extensions. Made obsolete by RFC 4366.
- RFC 3749: “Transport Layer Security Protocol Compression Methods”, specifies the framework for compression methods and the DEFLATE compression method.
- RFC 3943: “Transport Layer Security (TLS) Protocol Compression Using Lempel-Ziv-Stac (LZS)”.
- RFC 4132: “Addition of Camellia Cipher Suites to Transport Layer Security (TLS)”.
- RFC 4162: “Addition of SEED Cipher Suites to Transport Layer Security (TLS)”.
- RFC 4217: “Securing FTP with TLS”.
- RFC 4279: “Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)”, adds three sets of new cipher suites for the TLS protocol to support authentication based on pre-shared keys.

Extensions to TLS 1.1 include:

- RFC 4347: “Datagram Transport Layer Security” specifies a TLS variant that works over datagram protocols (such as UDP).
- RFC 4366: “Transport Layer Security (TLS) Extensions” describes both a set of specific extensions and a generic extension mechanism.
- RFC 4492: “Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)”.
- RFC 4507: “Transport Layer Security (TLS) Session Resumption without Server-Side State”.
- RFC 4680: “TLS Handshake Message for Supplemental Data”.
- RFC 4681: “TLS User Mapping Extension”.
- RFC 4785: “Pre-Shared Key (PSK) Ciphersuites with NULL Encryption for Transport Layer Security (TLS)”.
- RFC 5054: “Using the Secure Remote Password (SRP) Protocol for TLS Authentication”. Defines the TLS-SRP ciphersuites.
- RFC 5081: “Using OpenPGP Keys for Transport Layer Security (TLS) Authentication”, obsoleted by RFC 6091.

Extensions to TLS 1.2 include:

- RFC 5746: “Transport Layer Security (TLS) Renegotiation Indication Extension”.
- RFC 5878: “Transport Layer Security (TLS) Authorization Extensions”.
- RFC 6091: “Using OpenPGP Keys for Transport Layer Security (TLS) Authentication“.
- RFC 6176: “Prohibiting Secure Sockets Layer (SSL) Version 2.0”.
- RFC 6209: “Addition of the ARIA Cipher Suites to Transport Layer Security (TLS)”.

WHAT IS TLS?

http://en.wikipedia.org/wiki/Transport_Layer_Security

FROM THEORY TO PRACTICE

- RSA, DSA, ECDSA
- Diffie–Hellman, ECDH
- HMAC
- MD5, SHA1, SHA-2
- DES, 3DES, RC4, AES

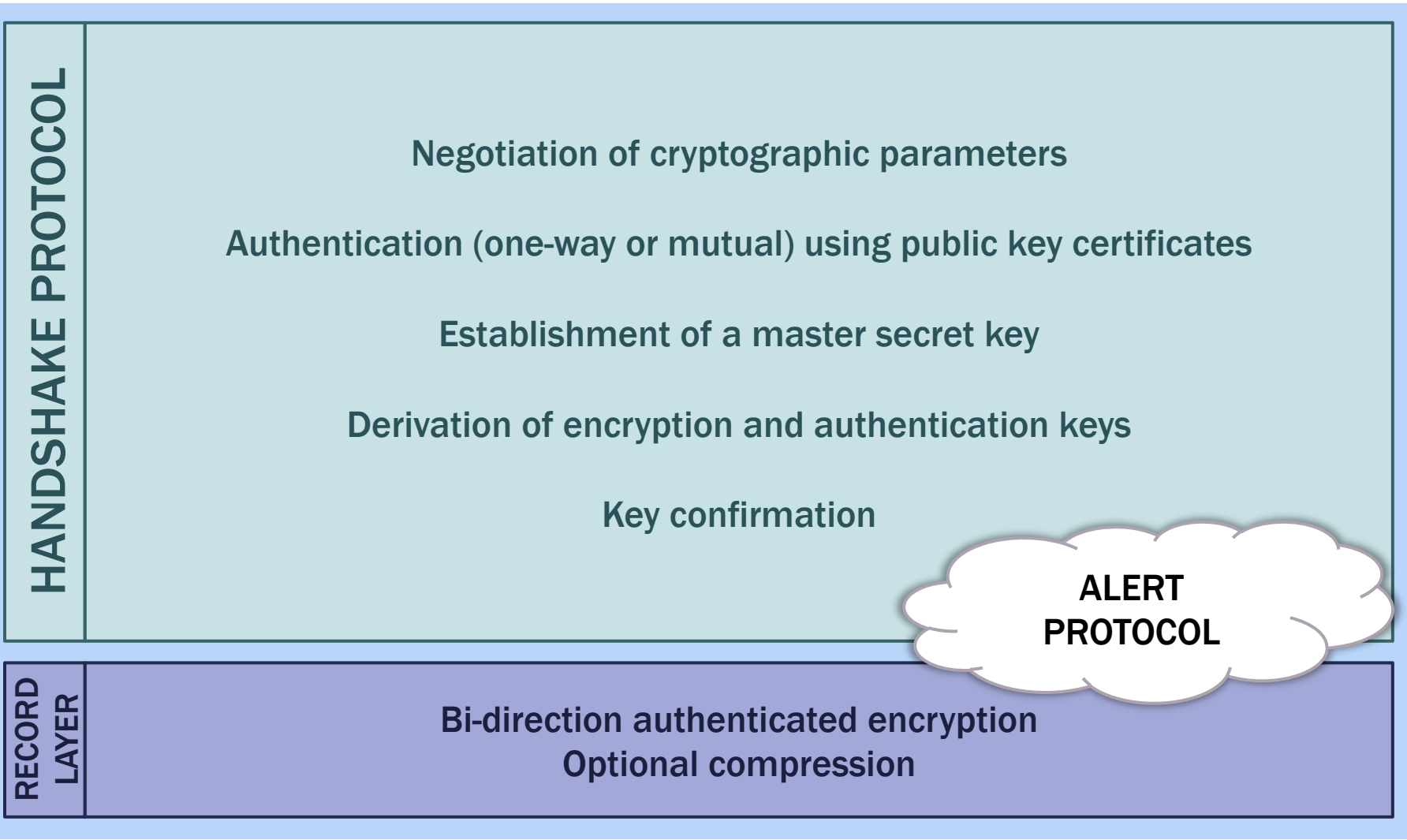
- Data structures
- Key derivation
- Encryption modes, IVs
- Padding
- Compresssion

- Alerts & errors
- Certification / revocation
- Negotiation
- Renegotiation
- Session resumption

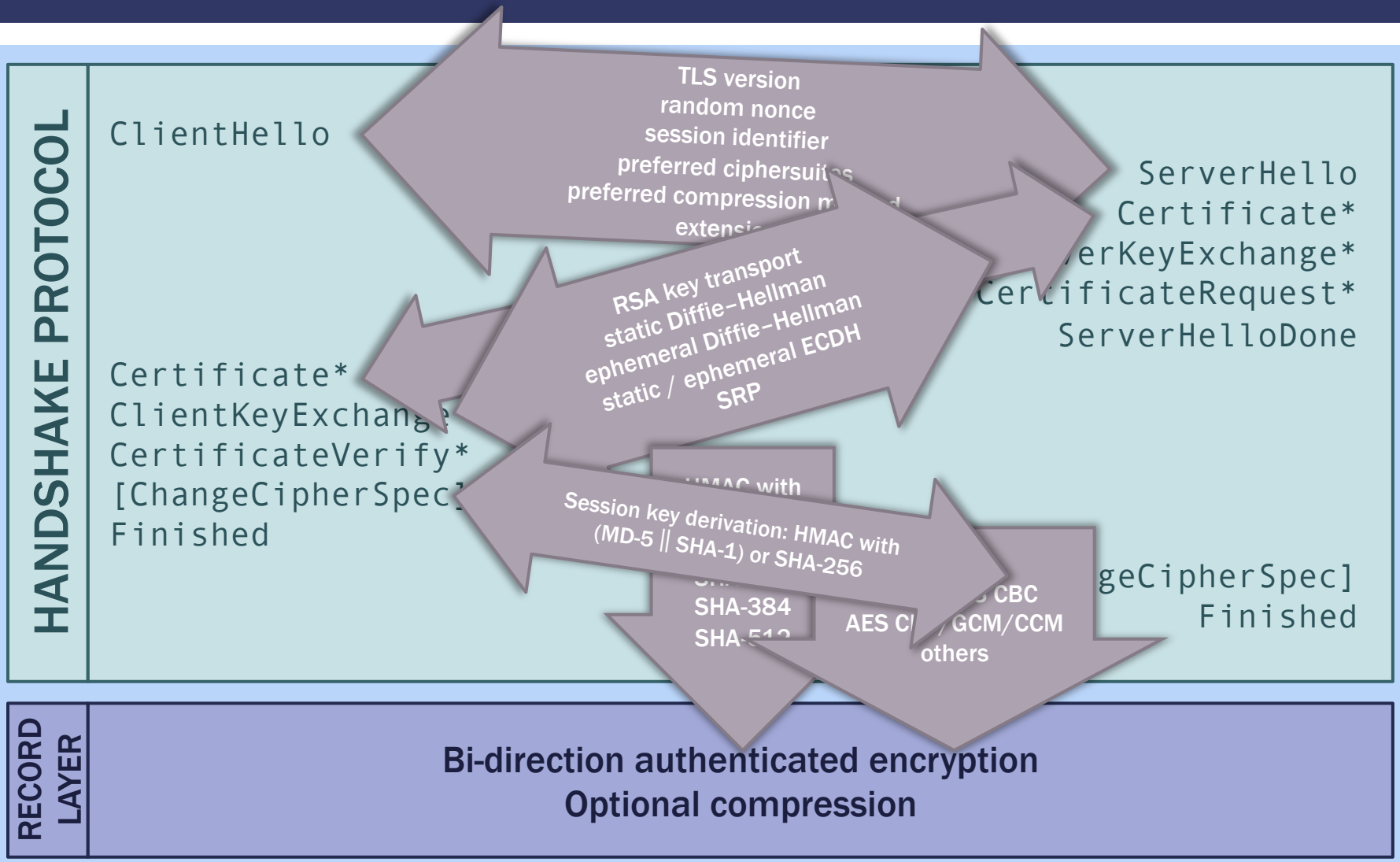
- OpenSSL
- GnuTLS
- SChannel
- Java JSSE

- Web browsers: Chrome, Firefox, IE, Safari
- Web servers: Apache, IIS, …
- Application SDKs

STRUCTURE OF TLS



STRUCTURE OF TLS



IS TLS SECURE?

IS TLS SECURE?

CORE CRYPTOGRAPHIC COMPONENTS

- Handshake protocol
  - secure authenticated key exchange protocol?
- Record layer
  - secure authenticated encryption channel?

ADDITIONAL PROTOCOL FUNCTIONALITY

- Alerts & errors?
- Certification?
- Renegotiation?
- Session resumption?

AUTHENTICATED KEY EXCHANGE PROTOCOLS

- Two parties aim to establish a shared secret in the presence of an active attacker who controls all communication and can potentially compromise certain secret values.

- Adversary's goal:
  - Given either the secret key of an uncompromised session or a random bitstring of the same length, decide which is the case.

- Various security models allow different secret values to be compromised:
  - Bellare–Rogaway 1993; Blake-Wilson–Johnson–Menezes 1995
  - Canetti–Krawczyk 2001
  - eCK 2007

PROVABLE SECURITY OF TLS HANDSHAKE PROTOCOL

- Classical result:
  - Signed Diffie–Hellman is a secure authenticated key exchange protocol.

HANDSHAKE LAYER

```
ClientHello                    -------->
                                                    ServerHello
                                                   Certificate*
                                             ServerKeyExchange*
                                            CertificateRequest*
                               <--------        ServerHelloDone
Certificate*
ClientKeyExchange
CertificateVerify*
[ChangeCipherSpec]
Finished                       -------->
                                             [ChangeCipherSpec]
                               <--------             Finished
```

- Does this mean that the TLS Handshake Protocol using signed DH is a secure AKE protocol?

PROVABLE SECURITY OF TLS HANDSHAKE PROTOCOL

- Classical result:
  - Signed Diffie–Hellman is a secure authenticated key exchange protocol.
- Does this mean that the TLS Handshake Protocol using signed DH is a secure AKE protocol?
- No ☹
- The Finished message — which has a recognizable format — is sent on the encrypted channel.
  - If the attacker is asked to decide between a real key and a random key, she can decrypt using the given key to see whether the plaintext looks like a valid Finished message or not.
- Truncated modified TLS with signed DHE is a secure AKE
  - Morissey, Smart, Warinschi; ASIACRYPT 2008
  - Gajek, Manulis, Pereira, Sadeghi, Schwenk; ProvSec 2008

PROVABLE SECURITY OF TLS RECORD LAYER PROTOCOL

- Security goal:
  - Authenticated encryption: integrity and confidentiality of ciphertexts
- Main technique:
  - MAC-then-encode-then-encrypt
- Security arguments:
  - Krawczyk; CRYPTO 2001:
    TLS with CBC encryption or stream ciphers is secure (IND-CPA, INT-CTXT), assuming random IVs and no padding
    - But IVs are not random! And there's padding (for CBC)!
  - Paterson, Ristenpart, Shrimpton; ASIACRYPT 2011:
    TLS with CBC encryption long MAC tags is secure length-hiding authenticated encryption (LHAE)

PROVABLE SECURITY OF TLS

- New security notion:
  - Authenticated and confidential channel establishment (ACCE)
- Jager, Kohlar, Schäge, Schwenk; CRYPTO 2012:
  - TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA is a secure ACCE protocol assuming
    - TLS PRF is secure
    - DSA is existentially unforgeable under chosen message attack
    - variant of oracle Diffie–Hellman assumption
    - record layer encryption is secure stateful length-hiding authenticated encryption (sLHAE)

ATTACKS!

- SSL 2.0: flawed in many ways
  - weak MAC
  - downgrade attacks
- SSL 3.0: alert message timing helps break RSA PKCSv1 (Bleichenbacher 1998)
- Implementation flaws
  - Weak Netscape PRNG (Goldberg & Wagner, 1995)
  - Debian OpenSSL entropy bug (2008)
- CBC encryption modes in record layer vulnerable
  - Bard 2004; Bard 2006
  - Rizzo & Duong "BEAST" attack 2011
  - mashups where attacker can inject data in same requests as sensitive user data; can be used to capture cookies
- Renegotiation in many applications vulnerable to plaintext injection (Ray & Dispensa 2009)
- Compression in record layer leaks side-channel information
  - Rizzo & Duong "CRIME" attack 2012
- More record layer vulnerabilities…?
- Non-browser TLS-reliant applications have poor certificate validation (CCS 2012)

THE GAP BETWEEN THEORY & PRACTICE

- RSA, DSA, ECDSA
- Diffie–Hellman, ECDH
- HMAC
- MD5, SHA1, SHA-2
- DES, 3DES, RC4, AES

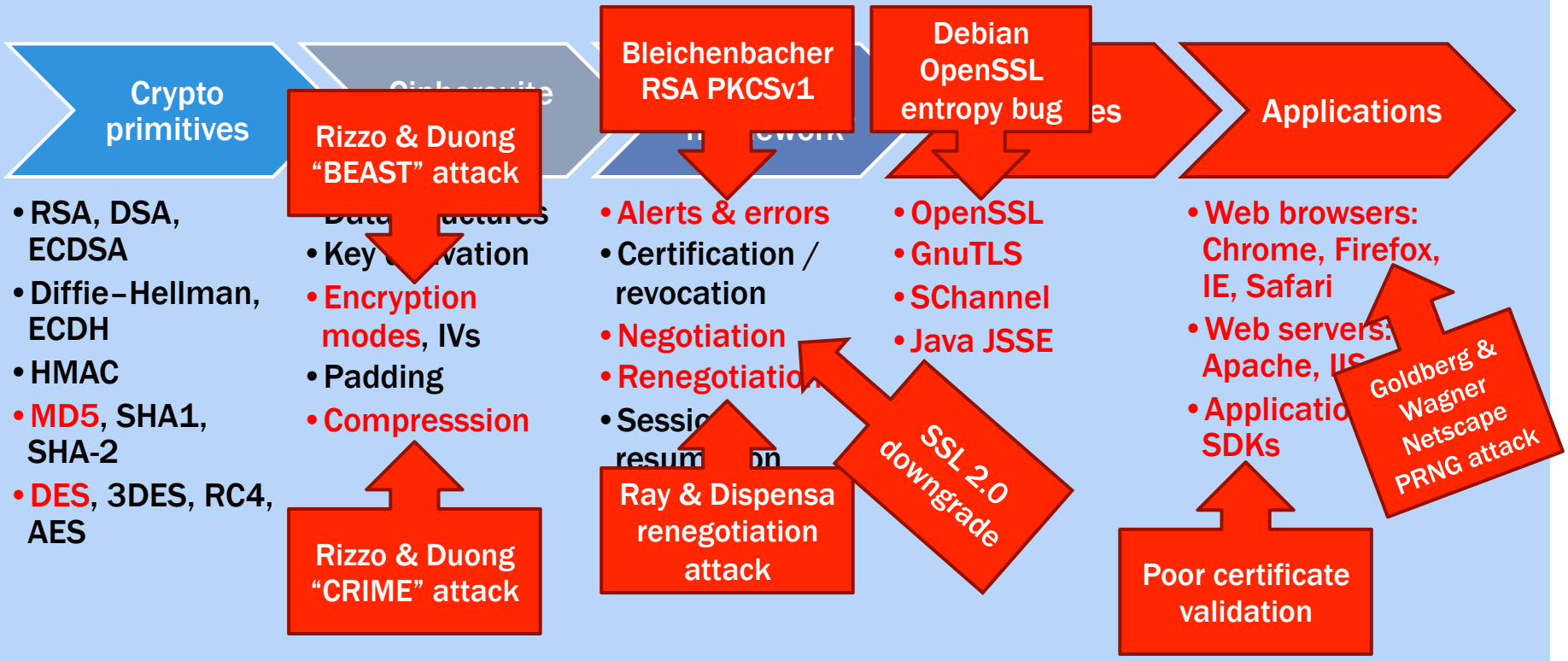
- Data structures
- Key derivation
- Encryption modes, IVs
- Padding
- Compresssion

- Alerts & errors
- Certification / revocation
- Negotiation
- Renegotiation
- Session resumption

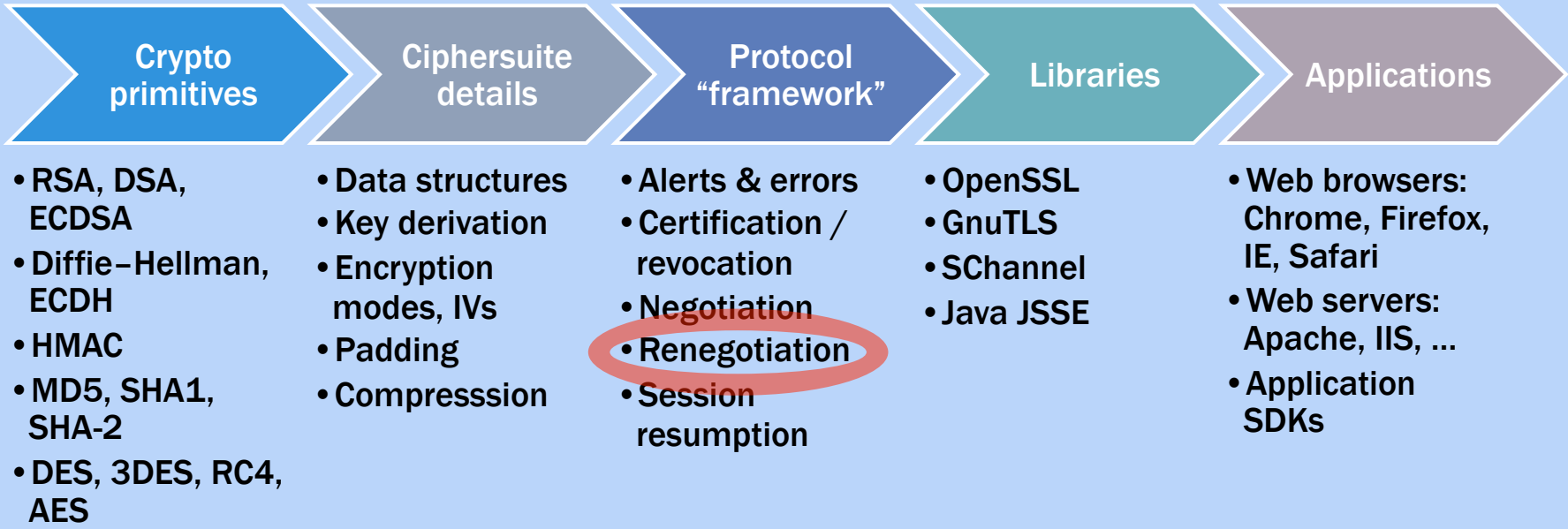
- OpenSSL
- GnuTLS
- SChannel
- Java JSSE

- Web browsers: Chrome, Firefox, IE, Safari
- Web servers: Apache, IIS, …
- Application SDKs

THE GAP BETWEEN THEORY & PRACTICE



THE GAP BETWEEN THEORY & PRACTICE



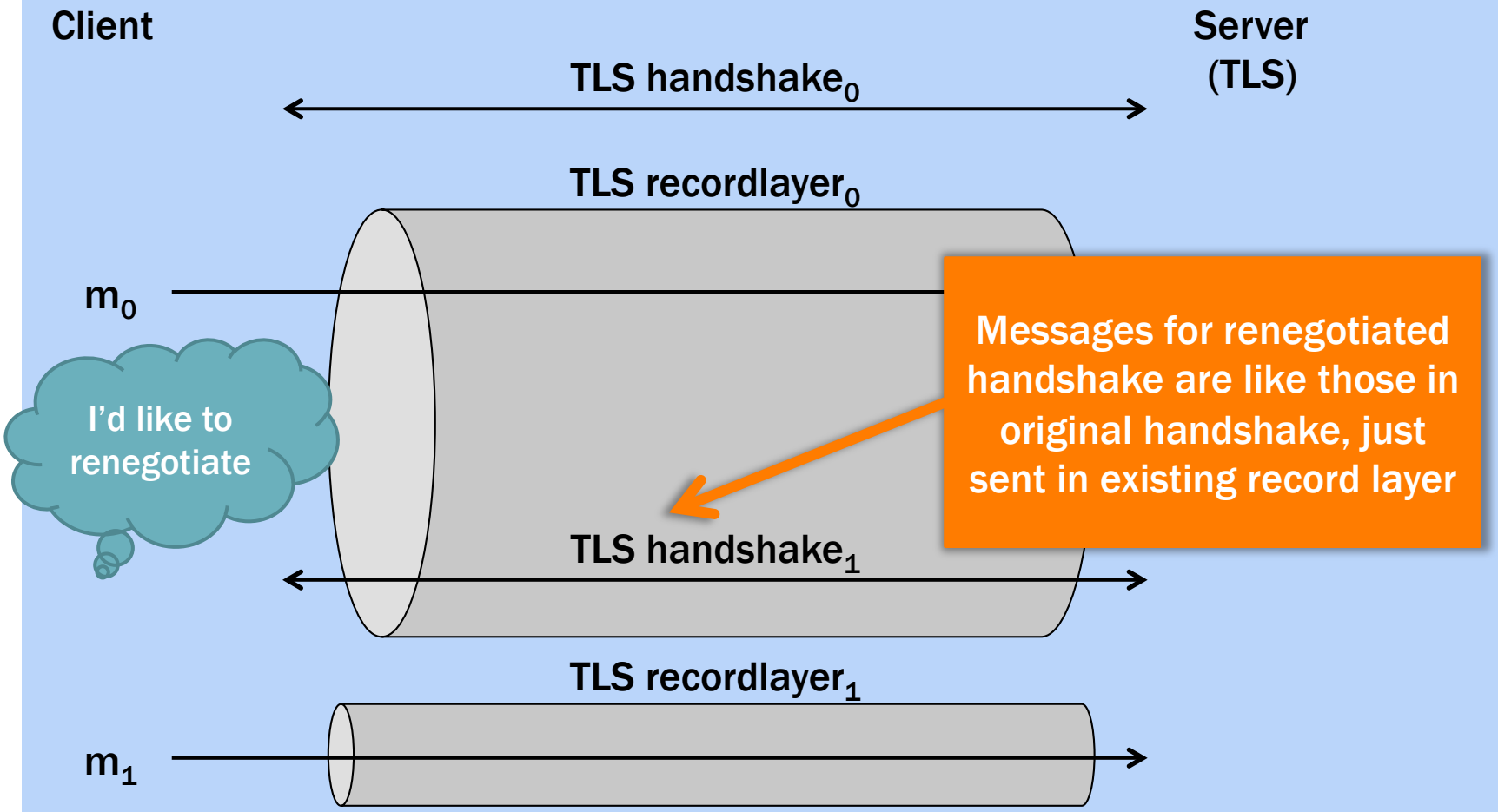
RENEGOTIATION

WHY RENEGOTIATE?

- Renegotiation allows parties in an established TLS channel to create a new TLS channel that continues from the existing one.

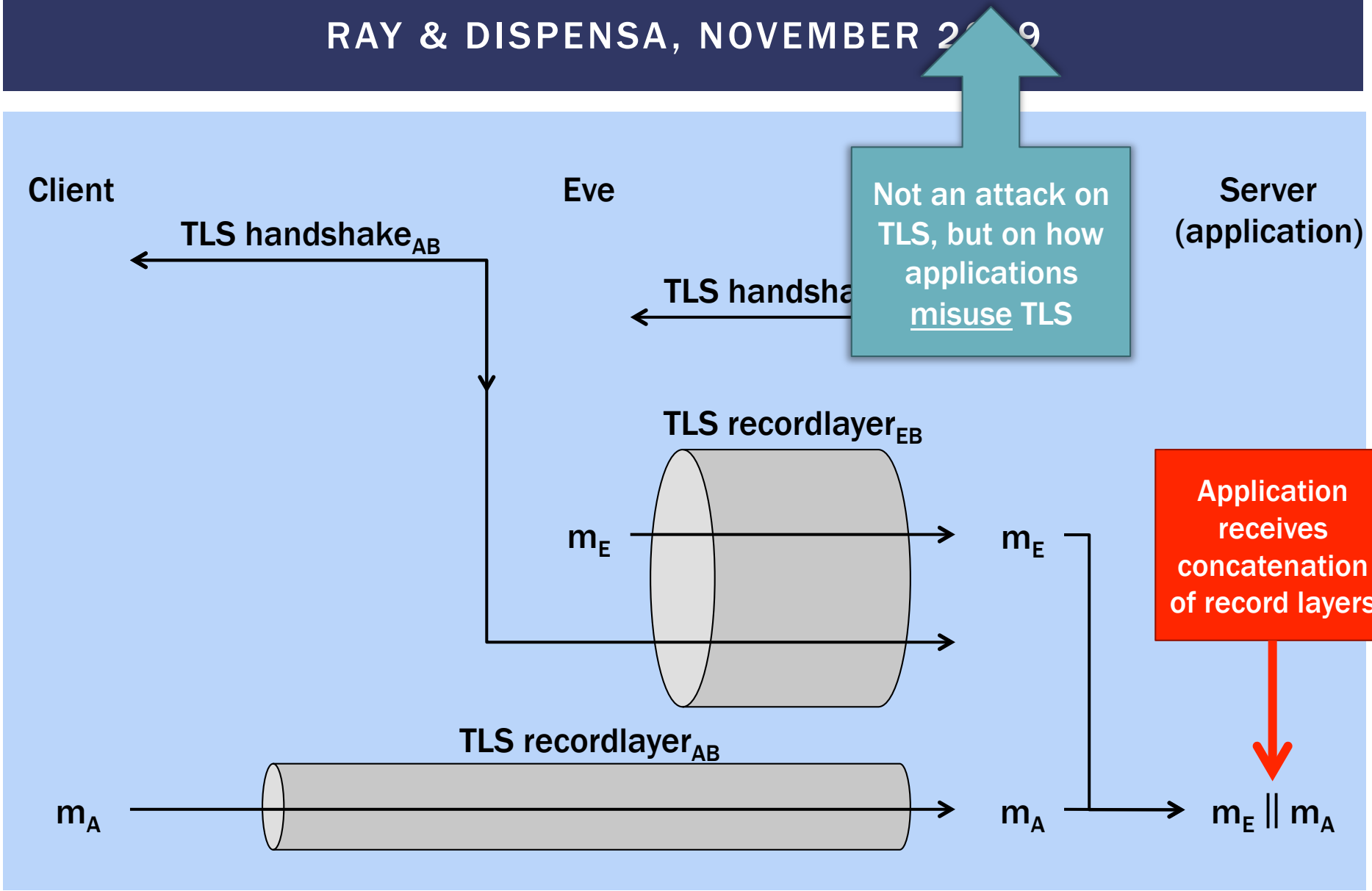
- Once you've established a TLS channel, why would you ever want to renegotiate it?
  - Change cryptographic parameters
  - Refresh encryption keys ("more perfect forward secrecy")
  - Change authentication credentials
  - Identity hiding for client
    1. Establish a one-way authenticated TLS session
    2. Renegotiate using mutual authentication.
       Since handshake messages are sent in the encrypted TLS channel, client's identity is kept private.

RENEGOTIATION IN TLS
(PRE-NOVEMBER 2009)



TLS RENEGOTIATION "ATTACK"

RAY & DISPENSA, NOVEMBER 2[illegible]9



EXAMPLE: HTTP INJECTION

- Attacker sends
  - m_E = "GET /orderPizza?deliverTo=123-Fake-St↩X-Ignore-This: "
- Client sends
  - m_A = "GET /orderPizza?deliverTo=456-Real-St↩Cookie: Account=1A2B"
- Server's web server receives
  - $m_E \| m_A$ = "GET /orderPizza?deliverTo=123-Fake-St↩
    X-Ignore-This: GET /orderPizza?deliverTo=456-Real-St↩
    Cookie: Account=1A2B"
  - X-Ignore-This: is an invalid header, so the rest of that line gets ignored.
  - The server's GET request is processed with the cookie supplied by the client.

WHY THE ATTACK WORKS

- The attack is not an attack on TLS security, but on how applications use TLS.
- Applications often see a TLS connection as a single socket and don't receive/process the data from the socket until it's all arrived.
- TLS allows renegotiation to take place at any time, including in the middle of an "incomplete" transmission.

VULNERABLE APPLICATION PROTOCOLS

- HTTPS without client certificates
- HTTPS with client certificates
  - TLS implementations don't by default check whether there is any connection between the client certificate in handshake_{EB} and handshake_{AB}
  - Applications only get the credentials from TLS socket when they query
- SMTPS with client certificates
- FTPS without client certificates
- more…

http://www.g-sec.lu/practicaltls.pdf

AFTERMATH

- Immediate workarounds:
  - Servers: disable renegotiation
  - Clients: … nothing
- RFC 5746: TLS Renegotiation Indication Extens
  - Client always includes in ClientHello message a renego
    - if n
    - if r … ake's Fin
  - Serve … E
    - if n
    - if r … ake's Fin
  - Alternative "signaling ciphersuite value" (SCSV) for clients worried about servers that may not understand extensions

Includes hash of all messages from … evious … shake

Does this fix the problem?

SECURITY OF TLS RENEGOTIATION

SECURITY OF TLS RENEGOTIATION

Recall: renegotiation “attack” is not an attack on TLS but on how applications misuse TLS

1. No need to fix TLS.
   Applications should just use TLS properly.

2. Fix TLS so that it’s hard to misuse.

RENEGOTIATION SECURITY

- Q: What property should a secure renegotiable protocol have?
- A: Whenever two parties successfully renegotiate, they are assured they have the exact same view of everything that happened previously.

TECHNICAL APPROACH

1. Extend authenticated and confidential channel establishment (ACCE) security model to include renegotiable, multi-phase protocols.

2. Define security notion for renegotiable protocols.
   - secure multi-phase ACCE
   - weakly secure renegotiable ACCE
   - secure renegotiable ACCE

3. Show that TLS without fixes does not satisfy security definition.

4. Show that TLS_DHE with fixes does satisfy security definition.
   - TLS_DHE is a weakly secure multi-phase ACCE
   - Every secure multi-phase ACCE combined with TLS fixes is a weakly secure renegotiable ACCE

5. Propose stronger fix.

ACCE SECURITY

AUTHENTICATED AND CONFIDENTIAL CHANNEL ESTABLISHMENT

- Extension of Bellare–Rogaway 1993 model for AKE
- Adversary controls all communications
- Parties have multiple sessions
  - with a "pre-accept stage" and a "post-accept stage" for each session
  - challenge bit $b_{i,s}$ for each session
- Queries
  - SendPre($\pi_{i,s}$, m): deliver message m to party i session s
  - Reveal($\pi_{i,s}$): reveal session key if pre-accept stage completed
  - Corrupt(i): reveal party i's long-term secret key
  - Encrypt($\pi_{i,s}$, m_0, m_1, len, head): encrypt either message m_0 or m_1 (based on bit $b_{i,s}$) using stateful length-hiding authenticated encryption
  - Decrypt($\pi_{i,s}$, c, head): if $b_{i,s} = 0$, return $\perp$; if $b_{i,s} = 1$ and c not a ciphertext output by Encrypt for the current state, output Dec(c)

ACCE SECURITY

AUTHENTICATED AND CONFIDENTIAL CHANNEL ESTABLISHMENT

- Adversary's goals:
  1. Violate authentication:
     - make some party i accept where its intended partner j is uncorrupted but has no matching session
  2. Violate ciphertext integrity or confidentiality:
     - guess bit $b_{i,s}$ in any session where intended partner j was uncorrupted and no Reveal query was issued for session or matching session
- Jager, Kohlar, Schäge, Schwenk; CRYPTO 2012:
  - TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA is a secure ACCE protocol assuming
    - TLS PRF secure
    - DSA existentially unforgeable under chosen message attack
    - variant of oracle Diffie–Hellman assumption
    - record layer is stateful length-hiding authenticated encryption

MULTI-PHASE ACCE

DEFINITION

- A session consists of an arbitrary number of phases.
  - Each phase has a pre-accept stage and a post-accept stage.
- Adjust model:
  - Reveal query
  - matching conversations
- Secure multi-phase ACCE:
  - Authentication: when a party successfully renegotiates a new phase, its partner has a phase with a matching handshake transcript.
  - Ciphertext integrity and confidentiality as before.

THEOREMS

- TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA is a secure multi-phase ACCE.
  - Under same assumptions as Jager et al.'s proof that it is a secure ACCE.
  - Same proof technique.
- TLS session resumption yields a secure multi-phase ACCE, assuming TLS is a secure ACCE.

SECURE RENEGOTIABLE ACCE

DEFINITION

- Secure renegotiable ACCE:
  - Authentication:
    - when a party successfully renegotiate a new phase, its partner has a phase with a matching handshake and record layer transcript

TLS

- TLS with or without RFC 5746 fixes is not a secure renegotiable ACCE.

WEAKLY SECURE RENEGOTIABLE ACCE

DEFINITION

- Weakly secure renegotiable ACCE:
  - Authentication:
    - when a party successfully renegotiate a new phase, its partner has a phase with a matching handshake and record layer transcript, *provided no previous phase's session key was revealed*

TLS

- TLS without fixes is not a weakly secure renegotiable ACCE.
- TLS with RFC 5746 fixes is a weakly secure renegotiable ACCE.

WEAKLY SECURE RENEGOTIABLE ACCE

TLS WITHOUT FIXES

- TLS without fixes is not a weakly secure renegotiable ACCE.
- Ray & Dispensa's attack means that client and server renegotiate with different views of previous handshakes

TLS WITH RFC 5746 FIXES

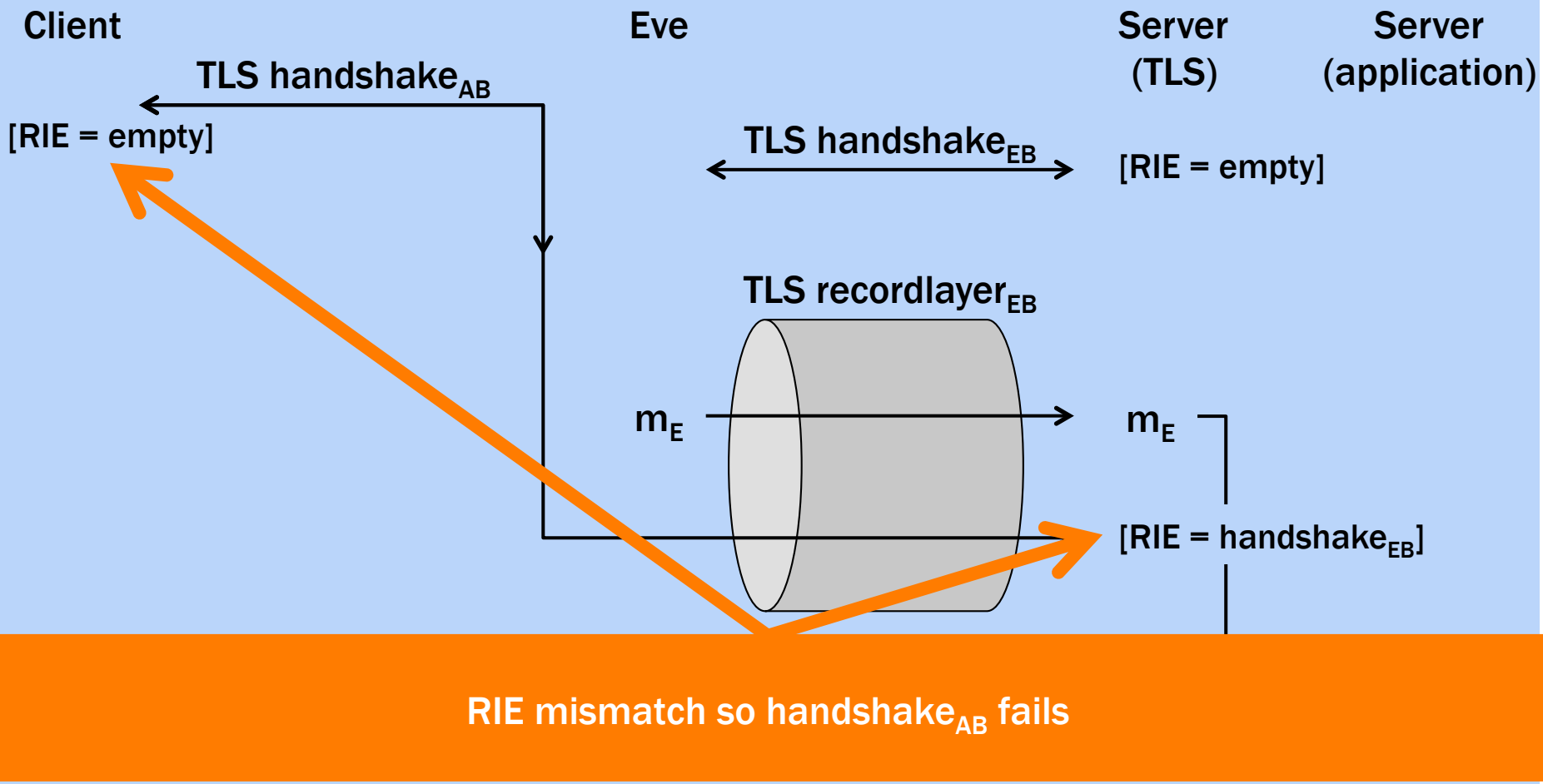
Theorem. If TLS with renegotiation indication extension (RIE) is a secure multi-phase ACCE, then it is also a weakly secure renegotiable ACCE.

Theorem. TLS_DHE with RIE is a secure multi-phase ACCE.

Corollary. TLS_DHE with RIE is a weakly secure renegotiable ACCE.

RENEGOTIATION ATTACK ON FIXED TLS?

RFC 5746 RENEGOTIATION INFORMATION EXTENSION



SECURE RENEGOTIABLE ACCE

TLS WITH RFC 5746 FIXES

- TLS with RIE is not a secure renegotiable ACCE.
- Adversary can reveal session key of current phase, change a message on the record layer, and parties will still renegotiate.
- This doesn't necessarily translate into an obvious attack.

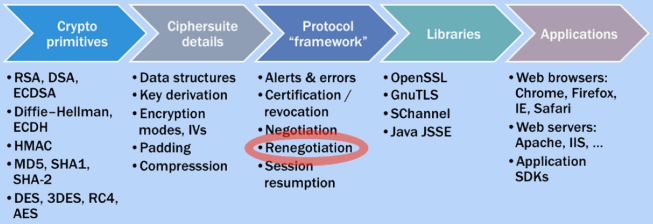
HOW TO MAKE TLS A SECURE RENEGOTIABLE ACCE

- Augment RIE with:
  - hash of all messages sent & received on the record layer in previous phase

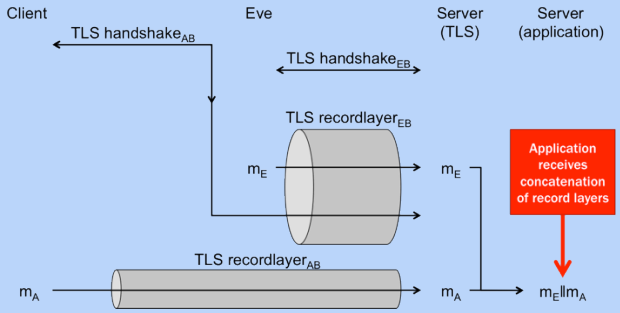
CONCLUSIONS

SUMMARY & CONCLUSIONS

1. TLS is more than just its core cryptographic protocol.



2. Many applications using TLS vulnerable to renegotiation attack.

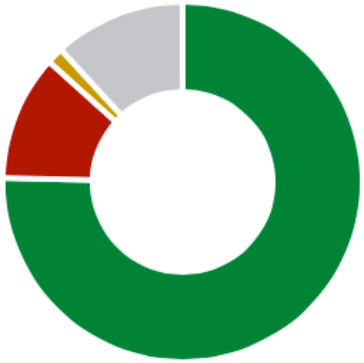


3. Including hashes of previous phases' handshake protocol transcripts <u>provably</u> detects renegotiation attacks (+ record layer transcripts for even stronger security).

ATTACK VULNERABILITY

SSL PULSE, AUGUST 10, 2012

Renegotiation Support



Secure renegotiation

139,785 75.8%

- 0.6 %

Insecure renegotiation

20,990 11.4%

+ 0.3 %

Both

1,609 0.9%

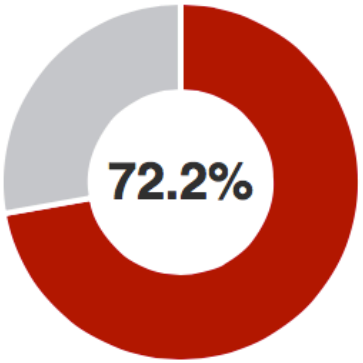
+ 0.0 %

No support

21,977 11.9%

+ 0.3 %

BEAST Attack



Sites that are vulnerable to the BEAST attack

133,115

+ 0.6 %

https://www.trustworthyinternet.org/ssl-pulse/

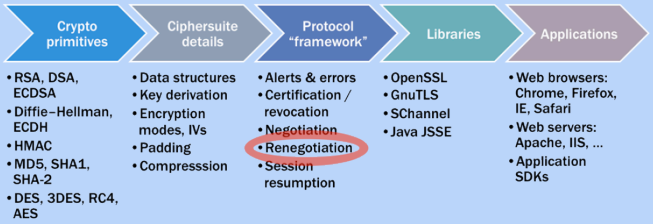
OPEN QUESTIONS

- Show that other TLS ciphersuites are secure LHAE/ACCE protocols.
  - e.g. RSA key transport with RC4 and SHA1 is the most widely used ciphersuite
- Relate CRIME attack to LHAE/ACCE security model.
- Extend ACCE model to cover one-way authenticated protocols.
  - Vast majority of TLS sessions are one-way, not mutually authenticated.
- Model additional TLS functionality:
  - certification
  - ciphersuite negotiation
  - modular framework for additional functionalities?
- Datagram TLS
- Consider other real-world protocols
  - SSH, Kerberos, …

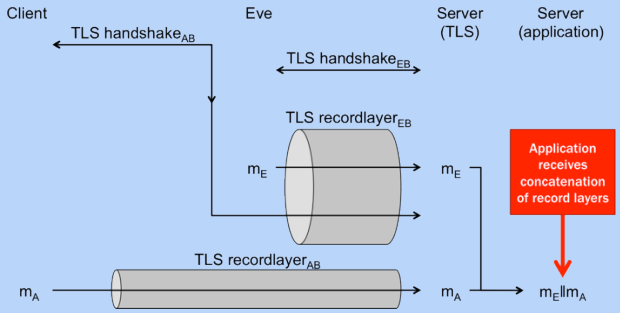
ON THE SECURITY OF TLS RENEGOTIATION

DOUGLAS STEBILA, QUEENSLAND UNIVERSITY OF TECHNOLOGY
JOINT WORK WITH FLORIAN GIESEN AND FLORIAN KOHLAR, RUHR-UNIVERSITÄT BOCHUM

1. TLS is more than just its core cryptographic protocol.



2. Many applications using TLS vulnerable to renegotiation attack.



3. Including hashes of previous phases' handshake protocol transcripts <u>provably</u> detects renegotiation attacks (+ record layer transcripts for even stronger security).