Practical Cryptographic Systems

Protocols III

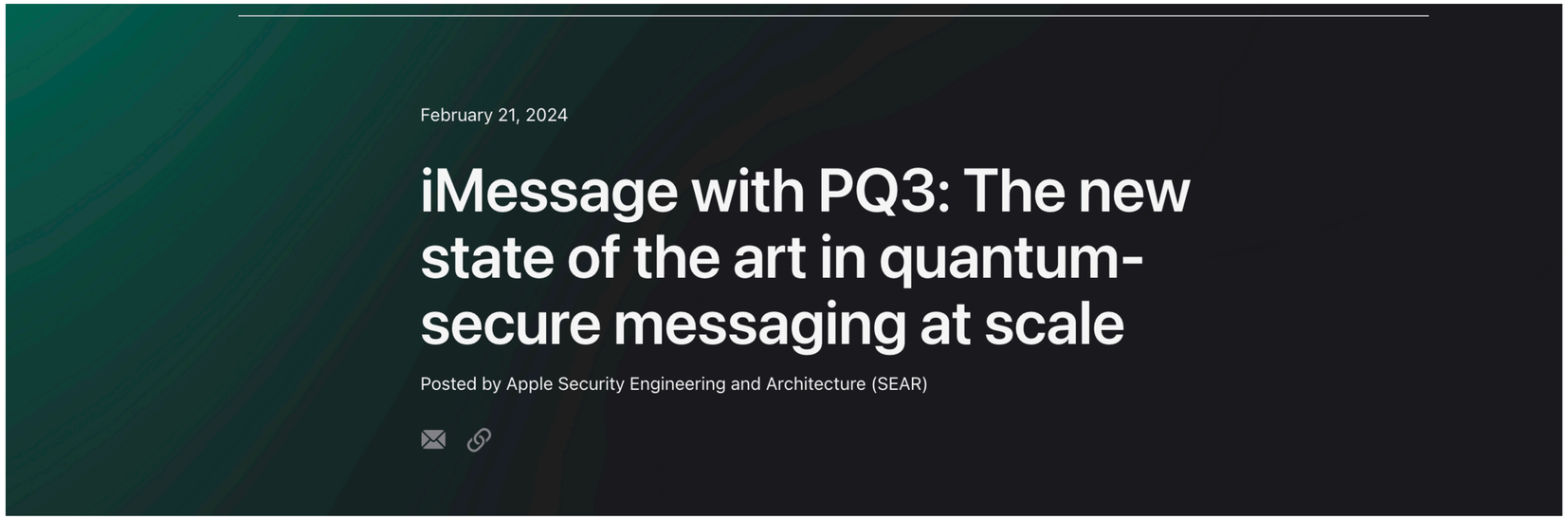
Instructor: Matthew Green

Housekeeping

- Reading assignment didn’t come out
-

News?

News?



February 21, 2024

iMessage with PQ3: The new state of the art in quantum-secure messaging at scale

Posted by Apple Security Engineering and Architecture (SEAR)

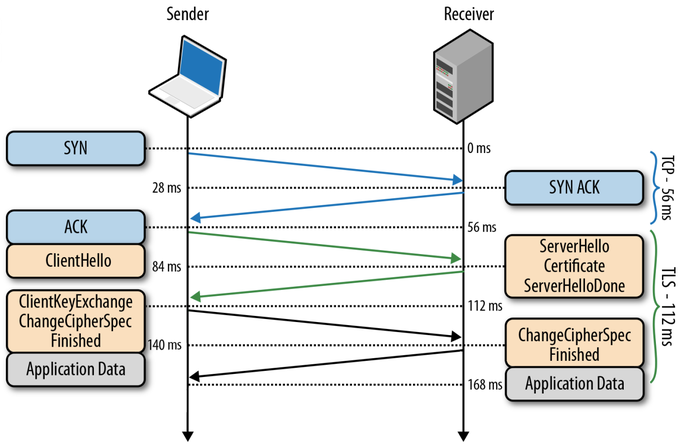
Today we are announcing the most significant cryptographic security upgrade in iMessage history with the introduction of PQ3, a groundbreaking post-quantum cryptographic protocol that advances the state of the art of end-to-end secure messaging. With compromise-resilient encryption and extensive defenses against even highly sophisticated quantum attacks, PQ3 is the first messaging protocol to reach what we call Level 3 security — providing protocol protections that surpass those in all other widely deployed messaging apps. To our knowledge, PQ3 has the strongest security properties of any at-scale messaging protocol in the world.

Protocols (definition)

- Definition
  - “A set of rules or procedures for transmitting data between electronic devices, such as computers”
  - “A security protocol (cryptographic protocol or encryption protocol) is an abstract or concrete protocol that performs a security-related function and applies cryptographic methods”

SSL/TLS

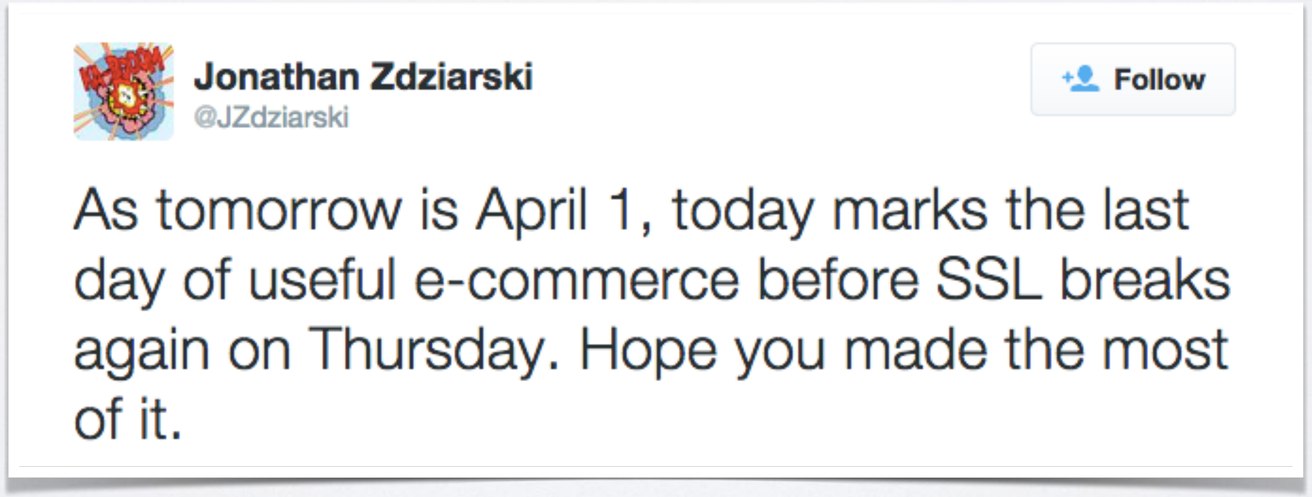
- Most important security protocol on the Internet
  - Allows secure connections between clients & servers
  - Current version: TLS 1.3 (RFC 8446)
    - (But browsers still support SSL 3
  - Not just web browsing!



How secure is TLS?

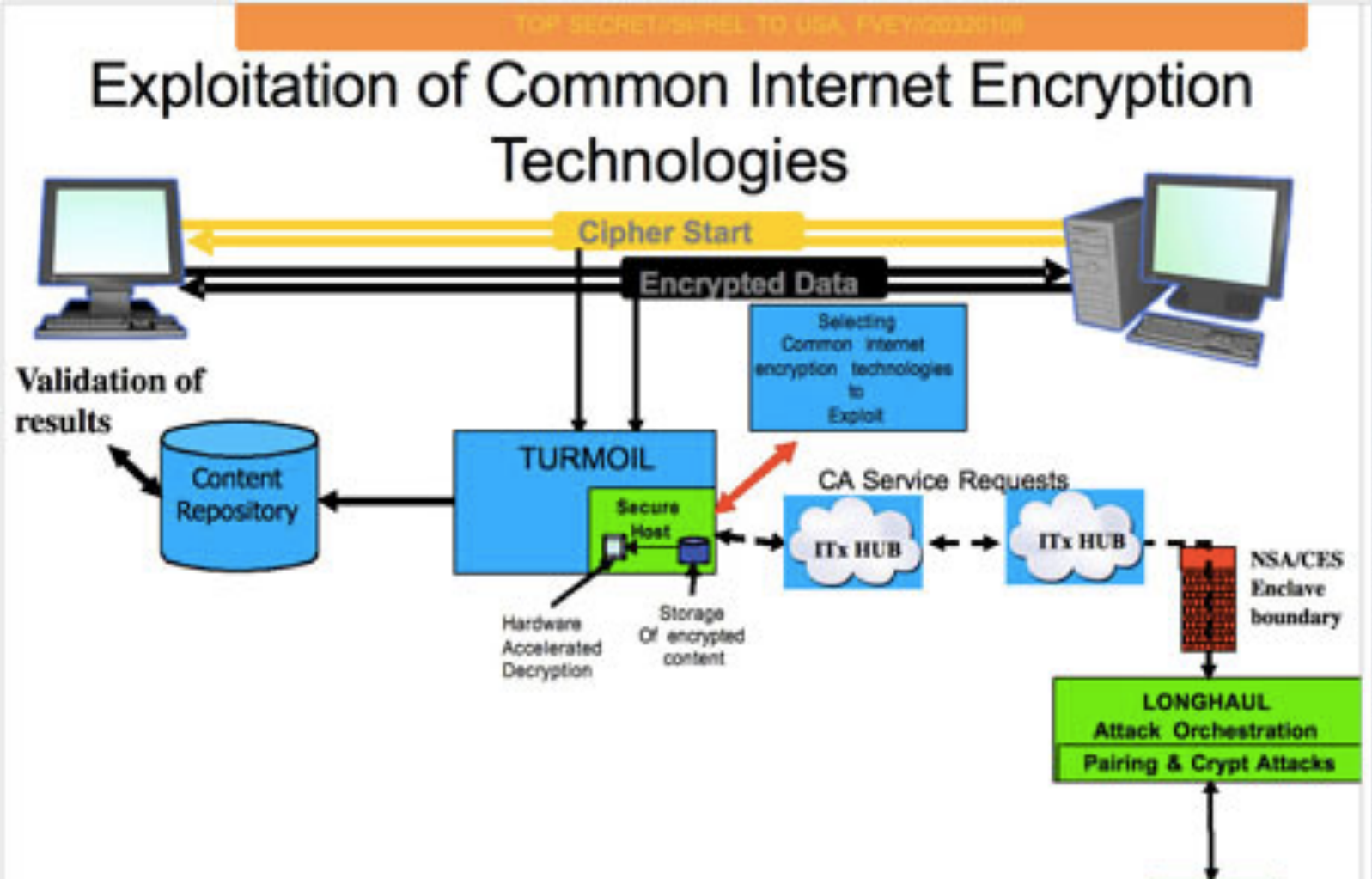
- **Many active attacks and implementation vulnerabilities**
  - Heartbleed, Lucky13, FREAK, CRIME, BEAST, RC4

In practice: most of these require substantial resources and can't be deployed at scale



How secure is TLS?

- But not all attacks…



What's wrong with TLS?

Quite a bit

- Many problems result from TLS's use of "*pre-historic cryptography*" (- Eric Rescorla)
  - CBC with Mac-then-Encrypt, bad use of IVs
  - RSA-PKCS#1v1.5 encryption padding
  - RC4
  - DH parameter generation
  - Horrifying backwards compatibility requirements

Quite a bit

- Many problems result from TLS's use of "*pre-historic cryptography*" (- Eric Rescorla)
  - CBC with Mac-then-Encrypt, bad use of IVs
  - RSA-PKCS#1v1.5 encryption padding
  - RC4
  - [illegible]
  - H[illegible]

Many of these flaws were 'known' at design time, but exploited by researchers only afterwards.

MAC-then-pad-then-Encrypt

- TLS MACs the record, then pads (in CBC), then enciphers
  - Obvious problem: padding oracles
  - Countermeasure(s):
    1. Do not distinguish padding/MAC failure
    2. "Constant-time" decryption

Unlucky for you: UK crypto-duo 'crack' HTTPS in Lucky 13 attack

OpenSSL patch to protect against TLS decryption boffinry

By **John Leyden** • **Get more from this author**

Posted in Security, 4th February 2013 16:58 GMT

BEAST

- Serious bug in TLS 1.0
- Allows complete decryption of CBC ciphertexts
- Use of predictable Initialization Vector (CBC residue bug)
  - Known since 2002, attack described by Bard in 2005 (*Bard was advised to focus on more interesting problems.*)
  - Nobody cared or noticed until someone implemented it

Solution in practice: RC4

:-(

(When RC4 is your solution,
you need a better problem)

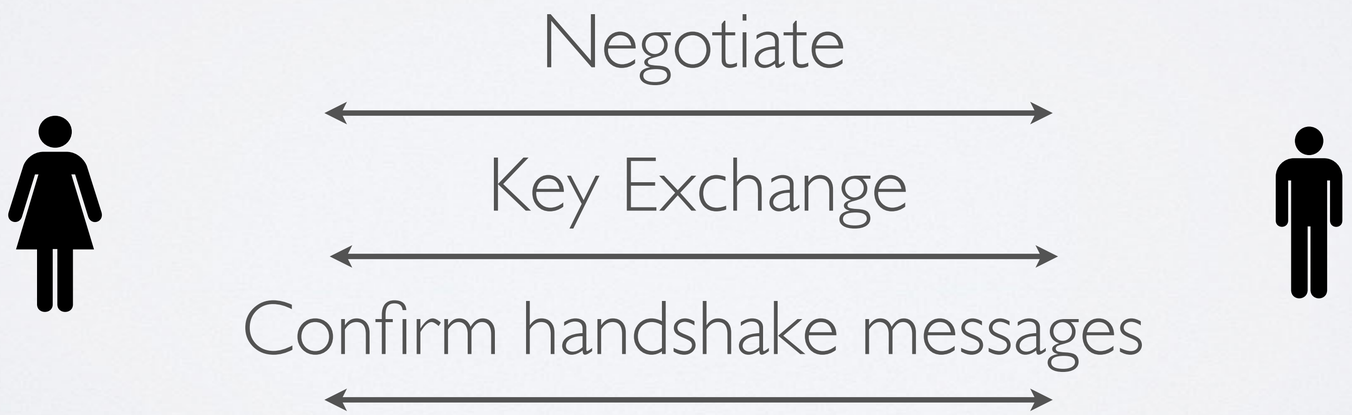
Compression (CRIME)

- Can't really blame the TLS designers for including it...
  - Blame cryptographers for not noticing it's still in use?
  - Blame cryptographers for pretending it would go away.
- We need a model for compression+encryption
  - Clearly this can't be semantically secure
  - *But how much weaker? Can we quantify?*

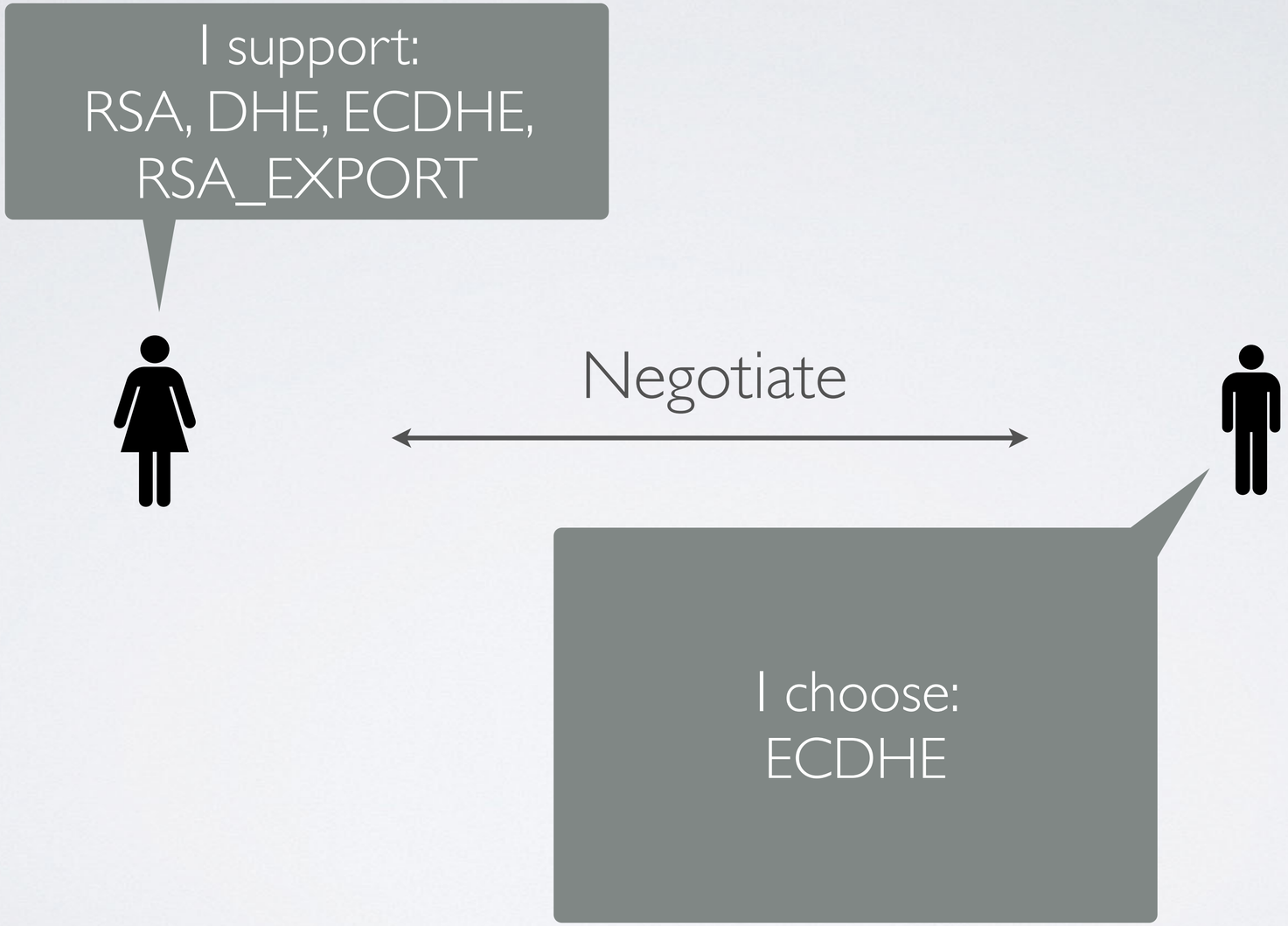
Protocol Design

Example: Negotiation

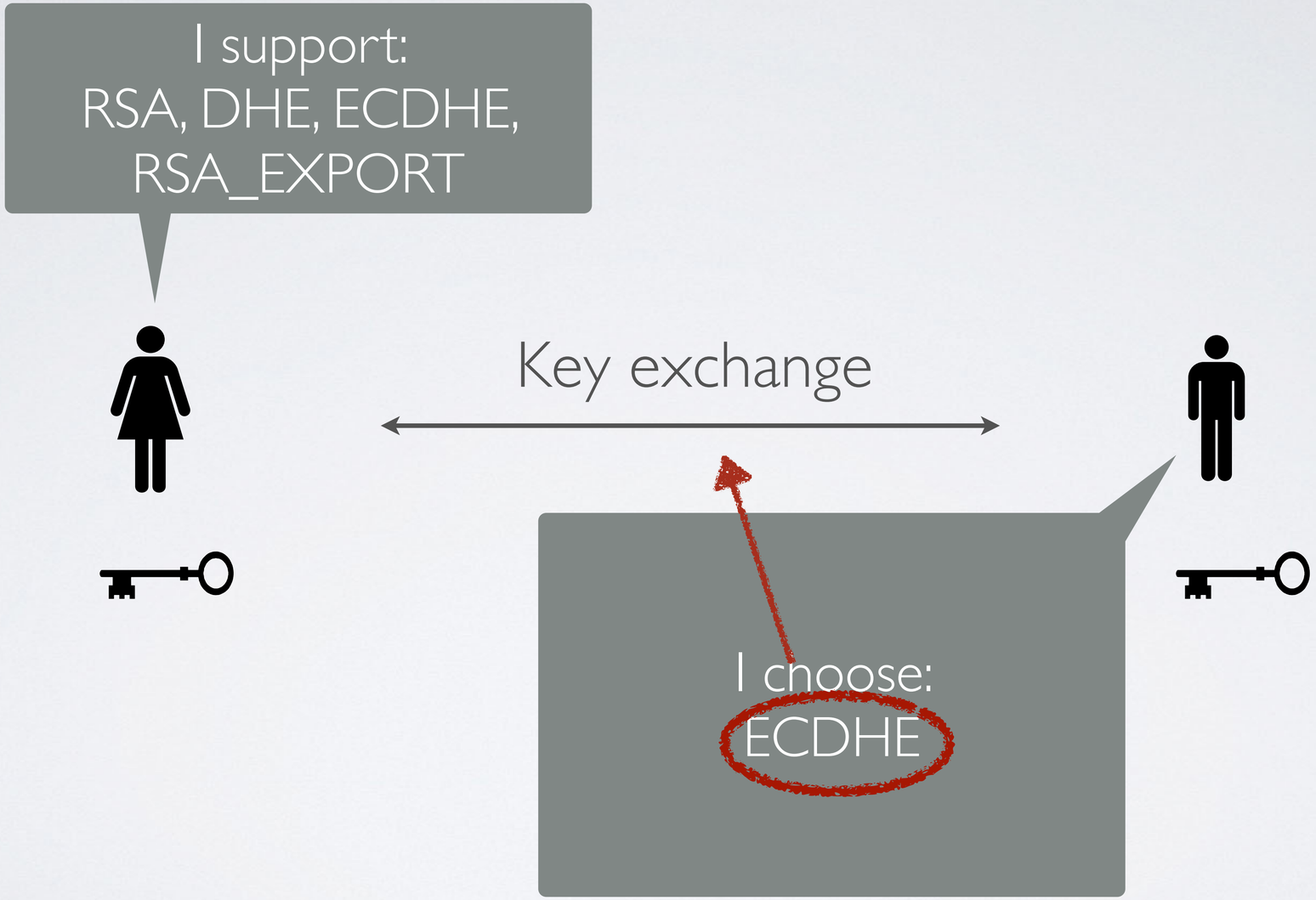
Each TLS handshake begins with a cipher suite negotiation that determines which key agreement protocol (etc.) will be used.



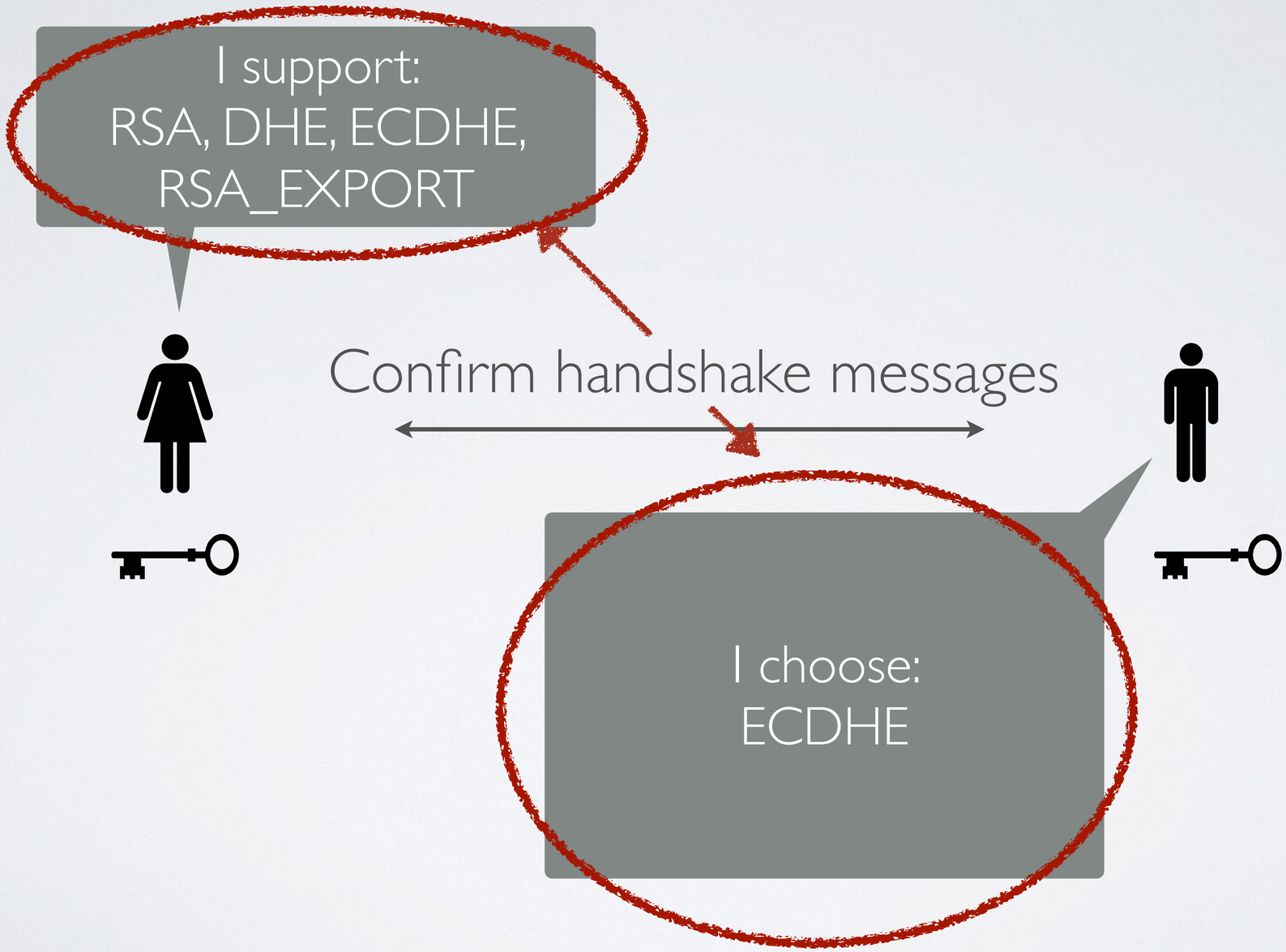
Ciphersuite Negotiation



Ciphersuite Negotiation



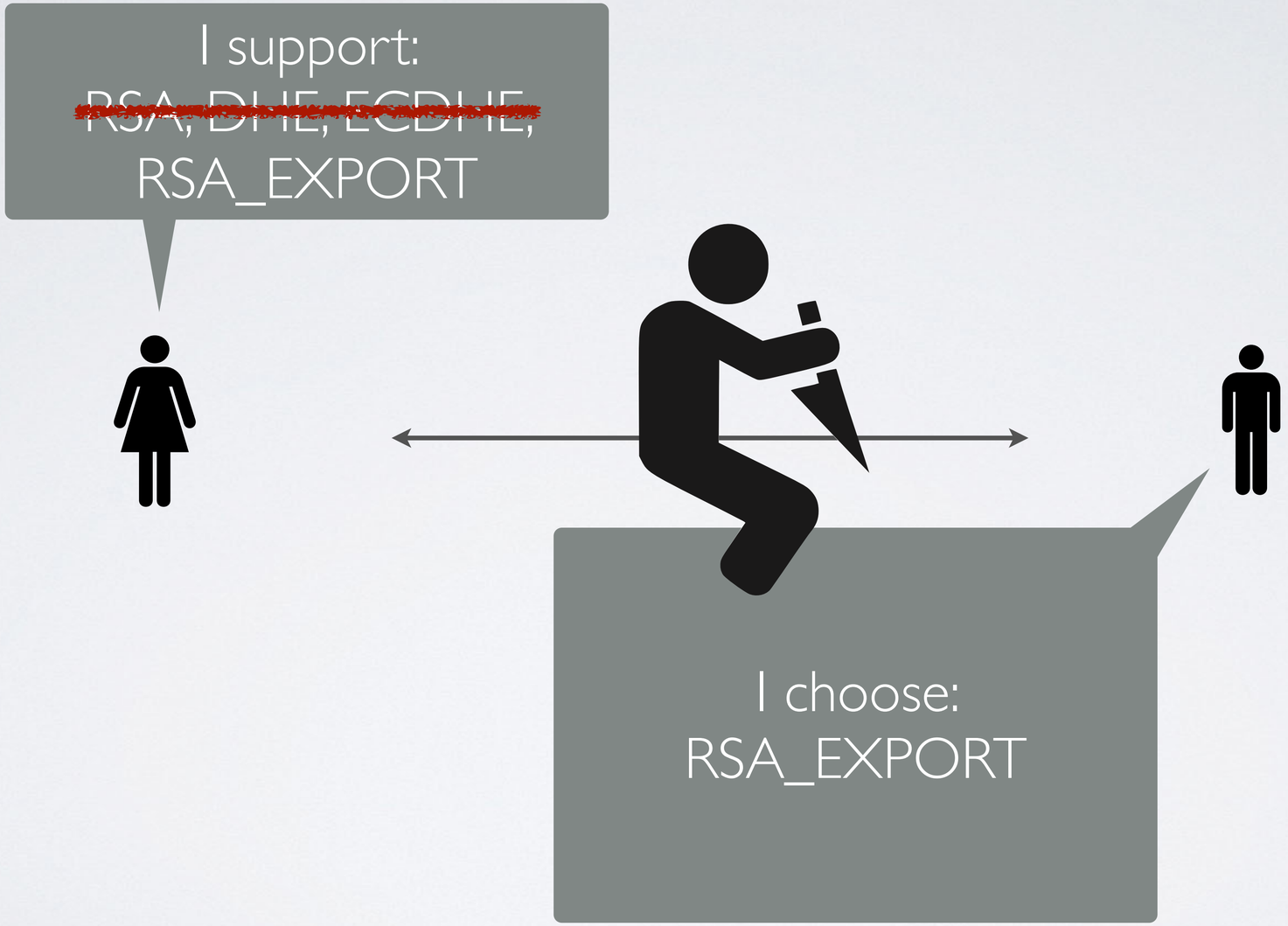
Ciphersuite Negotiation



MITM Negotiation

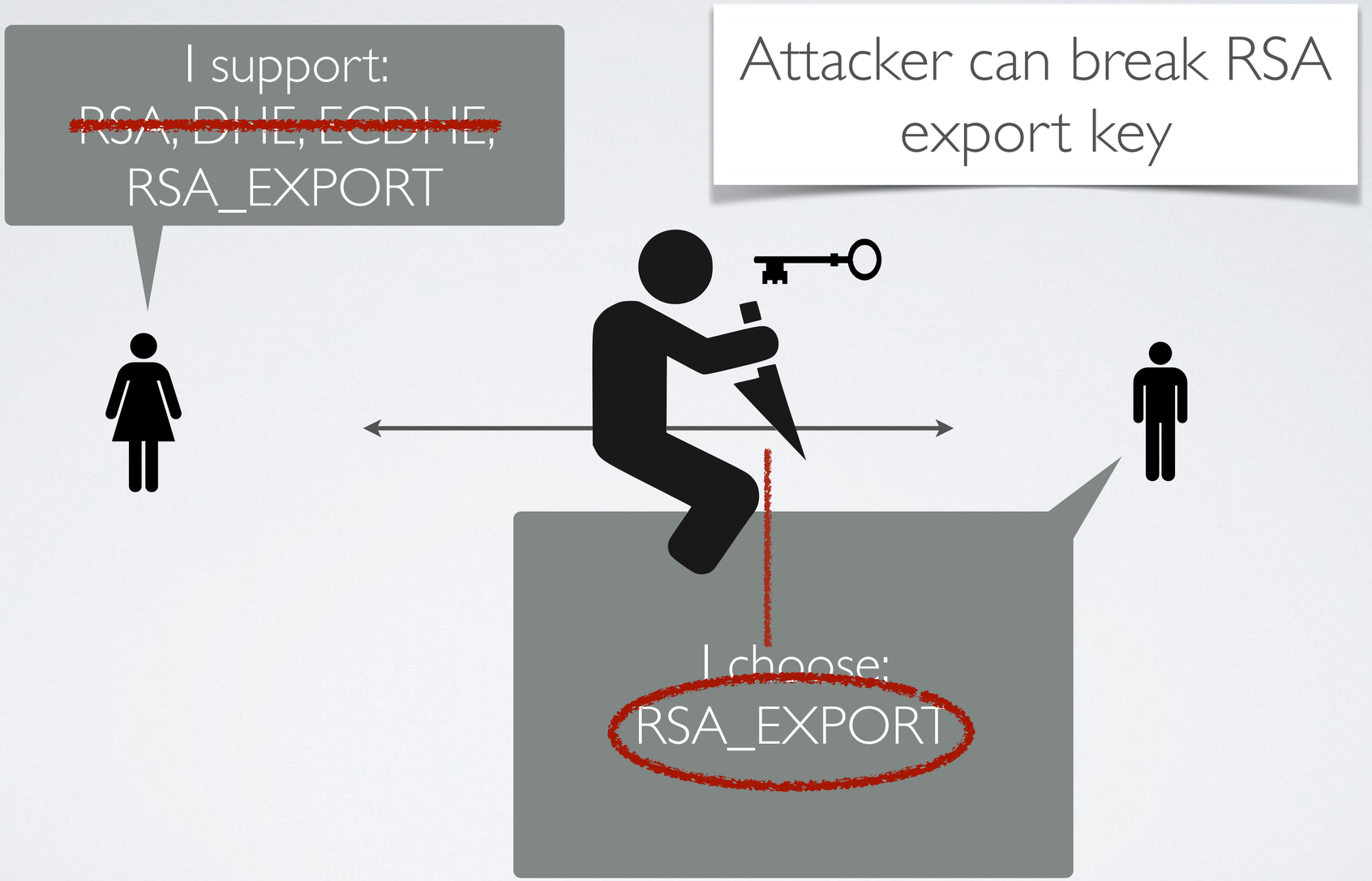
MITM Negotiation

MITM Negotiation

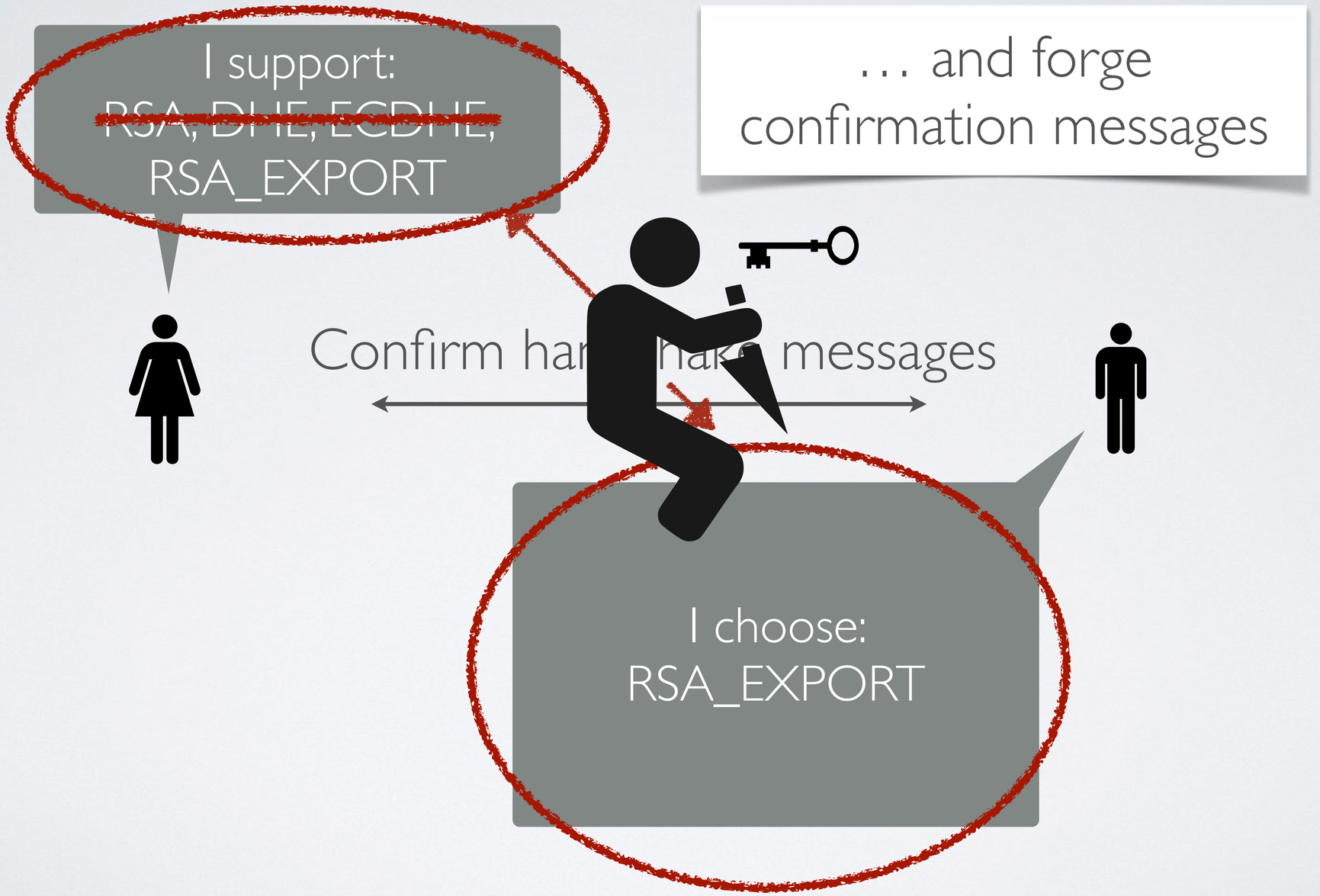


MITM Negotiation

MITM Negotiation



MITM Negotiation



MITM Negotiation

As of Mar '15,
30+% of TLS hosts supported
export suites!

MITM Negotiation

I support:
RSA_DHE_EGDHE

Solution:

Modern clients won't offer broken cipher suites like RSA_EXPORT

(unless they're wget or curl!)

I choose:
A_EXPORT

As of Mar '15,
30+% of TLS hosts supported export suites!

Question

Is it sufficient for the client to support only “strong” ciphersuites, even if the server supports weak ones?

Question

Is it sufficient for the client to support only “strong” ciphersuites, even if the server supports weak ones?

- Let ***A*** be the set of KA protocols supported by Client
  Let ***B*** be the set of KA protocols supported by Server

- If each KA protocol in $A \cap B$ is a secure KA protocol, is the TLS handshake secure

TLS for cryptographers

- In CRYPTO 2012 (!) we saw the first paper to successfully analyze TLS-DHE [Jager *et al.]*
- In CRYPTO 2013 a random-oracle analysis of the TLS-RSA handshake [Krawczyk *et al.]*
- In CRYPTO 2014 an automated analysis of the full handshake, under a new security model [Bhargavan *et al.]*

TLS for cryptographers

We do not model ciphersuite negotiation/renegotiation, nor session resumption.

- In CRYPTO 2012 (!) we saw the first paper to successfully analyze TLS-DHE [Jager *et al.]*
- In CRYPTO 2013 a random-oracle analysis of the TLS-RSA handshake [Krawczyk *et al.]*
- In CRYPTO 2014 an automated analysis of the full handshake, under a new security model [Bhargavan *et al.]*

TLS for cryptographers

- In CRYPTO 2012 (!) we saw the first paper to successfully analyze TLS-DHE [Jager *et al.]*
- In CRYPTO 2013 a random-oracle analysis of the TLS-RSA handshake [Krawczyk *et al.]*
- In CRYPTO 2014 an automated analysis of the full handshake, under a new security model [Bhargavan *et al.]*

Theorem

- **Bhargavan *et al*. theorem statement:**
  Let ***A*** be the set of KA protocols supported by Client
  Let ***B*** be the set of KA protocols supported by Server

  If each KA protocol in $A \cup B$ is a secure KA protocol & there exist PRFs, then the TLS handshake is a secure KA protocol.

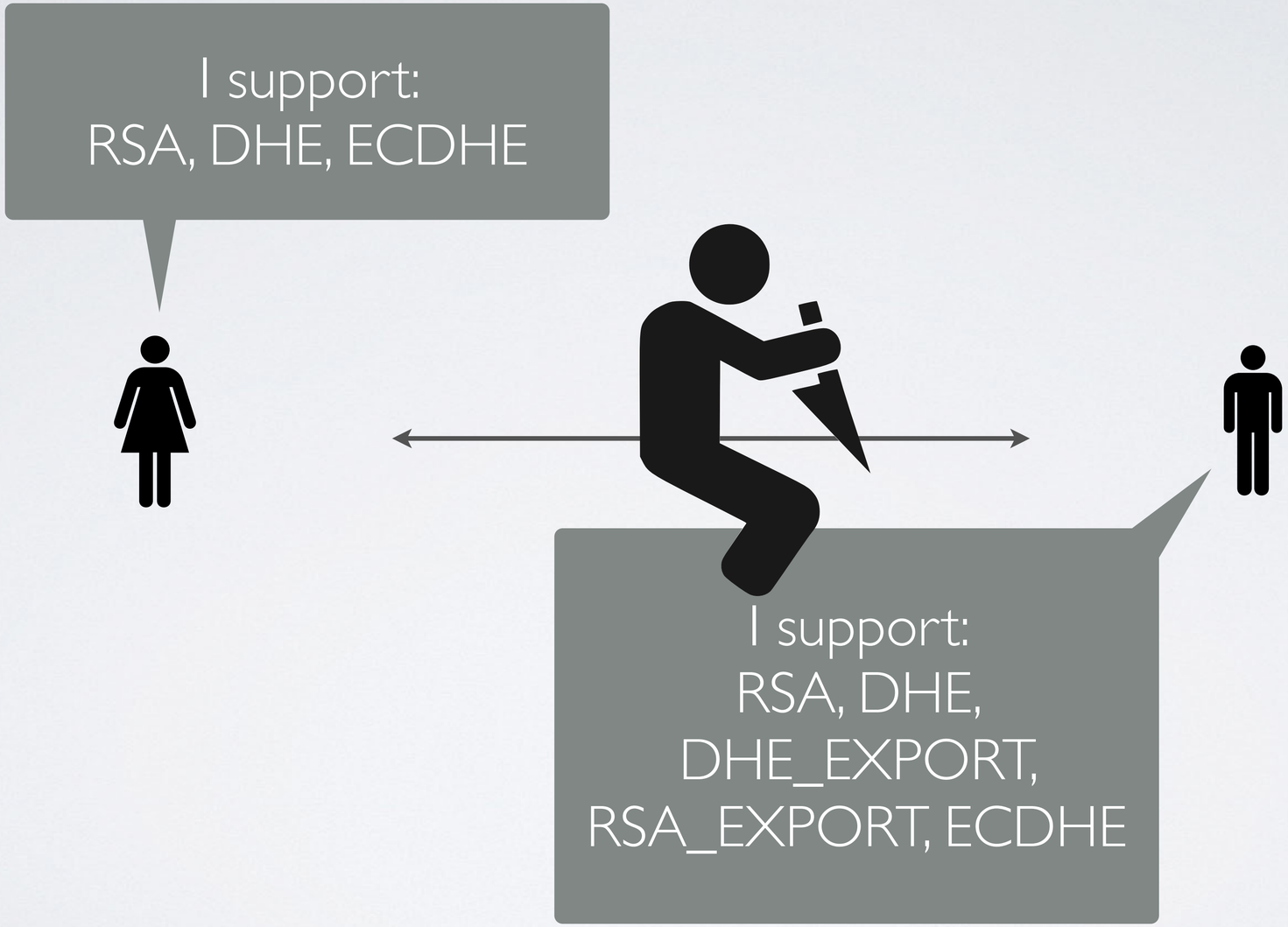
Theorem

- **Bhargavan *et al*. theorem statement:**
  Let ***A*** be the set of KA protocols supported by Client
  Let ***B*** be the set of KA protocols supported by Server

  If each KA protocol in $A \cup B$ is a secure KA protocol & there exist PRFs, then the TLS handshake is a secure KA protocol.

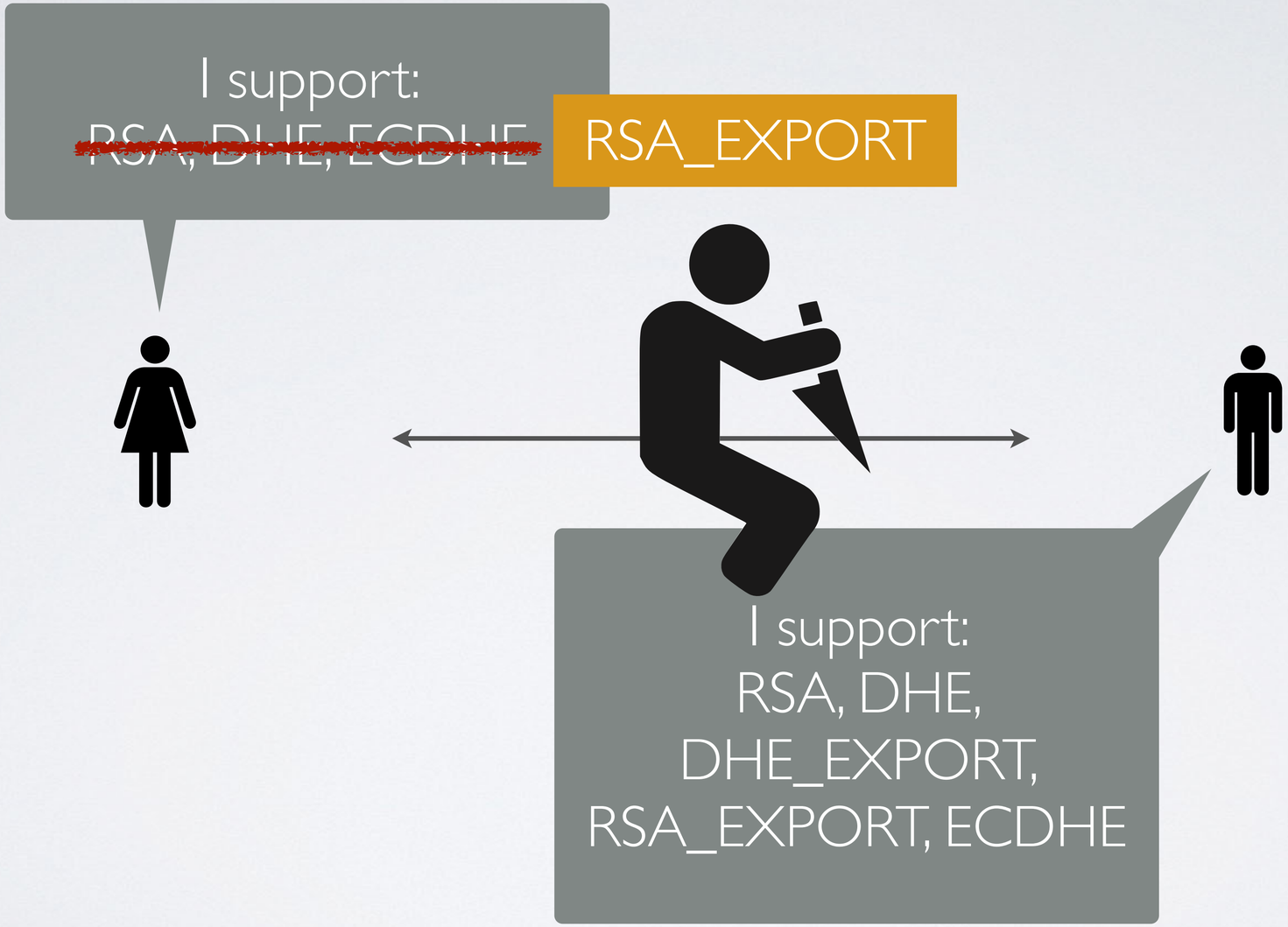
TLS design/deployment assumes this would be $A \cap B$!

Example 2: Negotiation



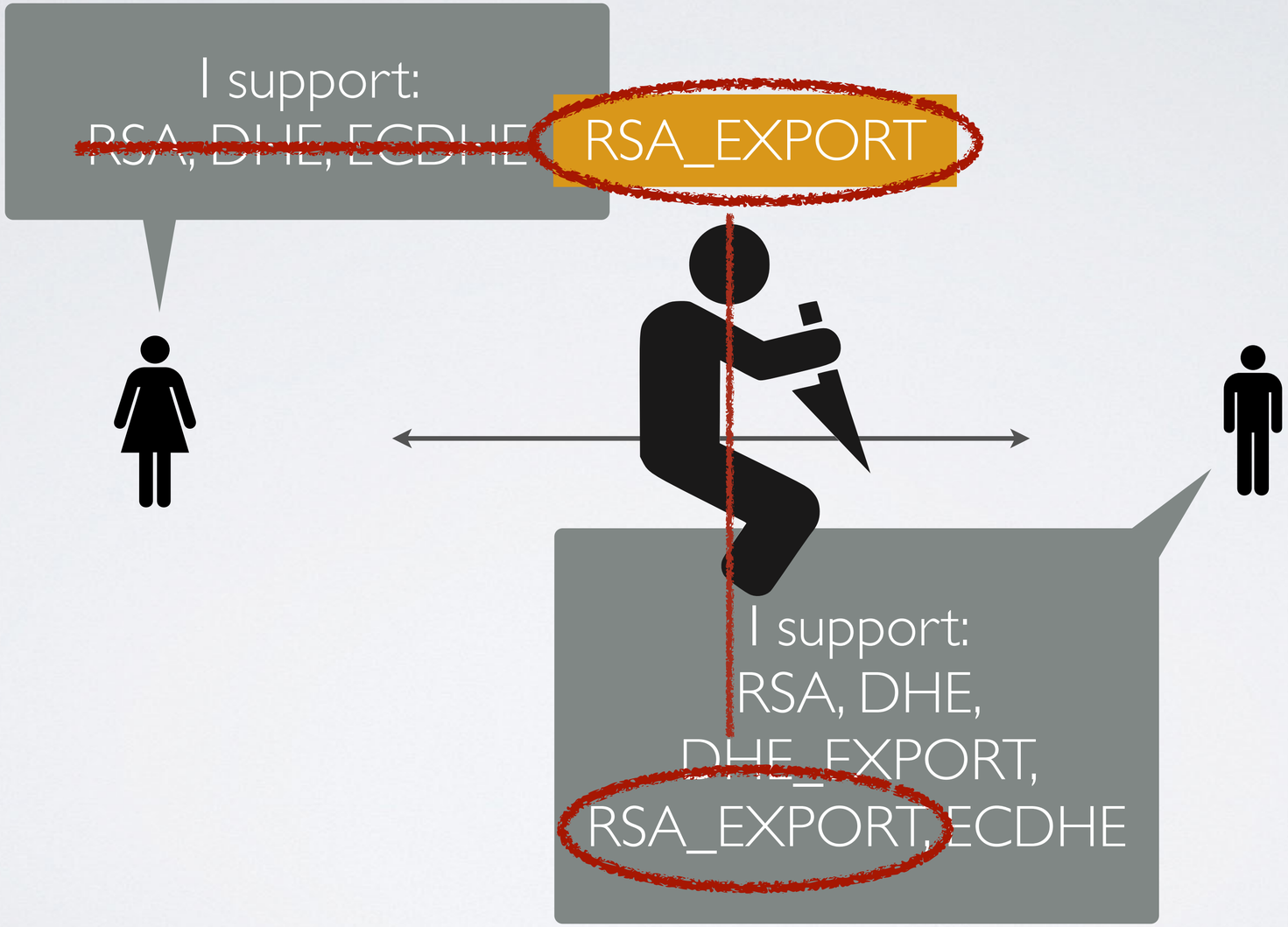
Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

Example 2: Negotiation



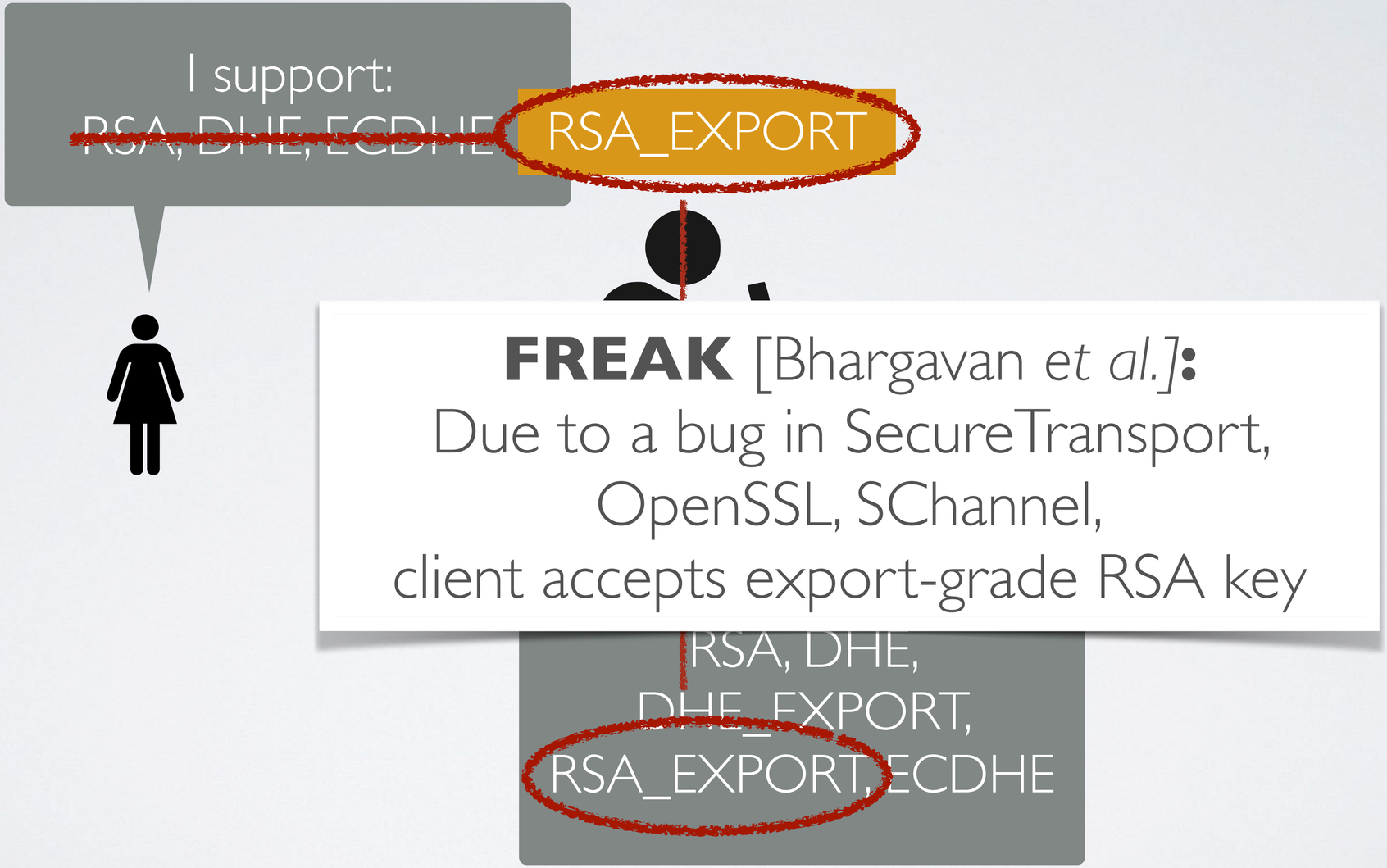
Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

Example 2: Negotiation

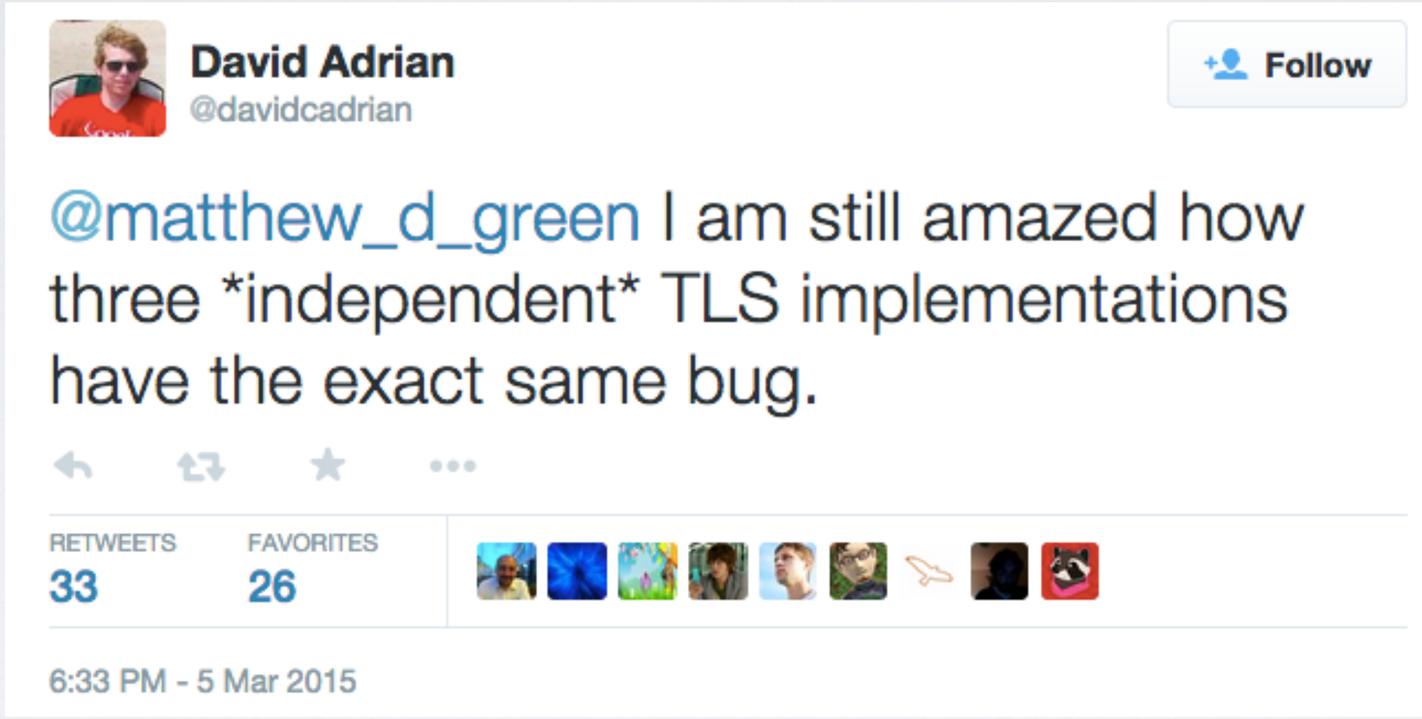


Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

Example 2: Negotiation



Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010



Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

Example 2: Negotiation

Solution: Fix implementations

Patch OpenSSL, SecureTransport, SChannel so they will recognize an RSA export key exchange message, barf

(patches rolled out March 2015)

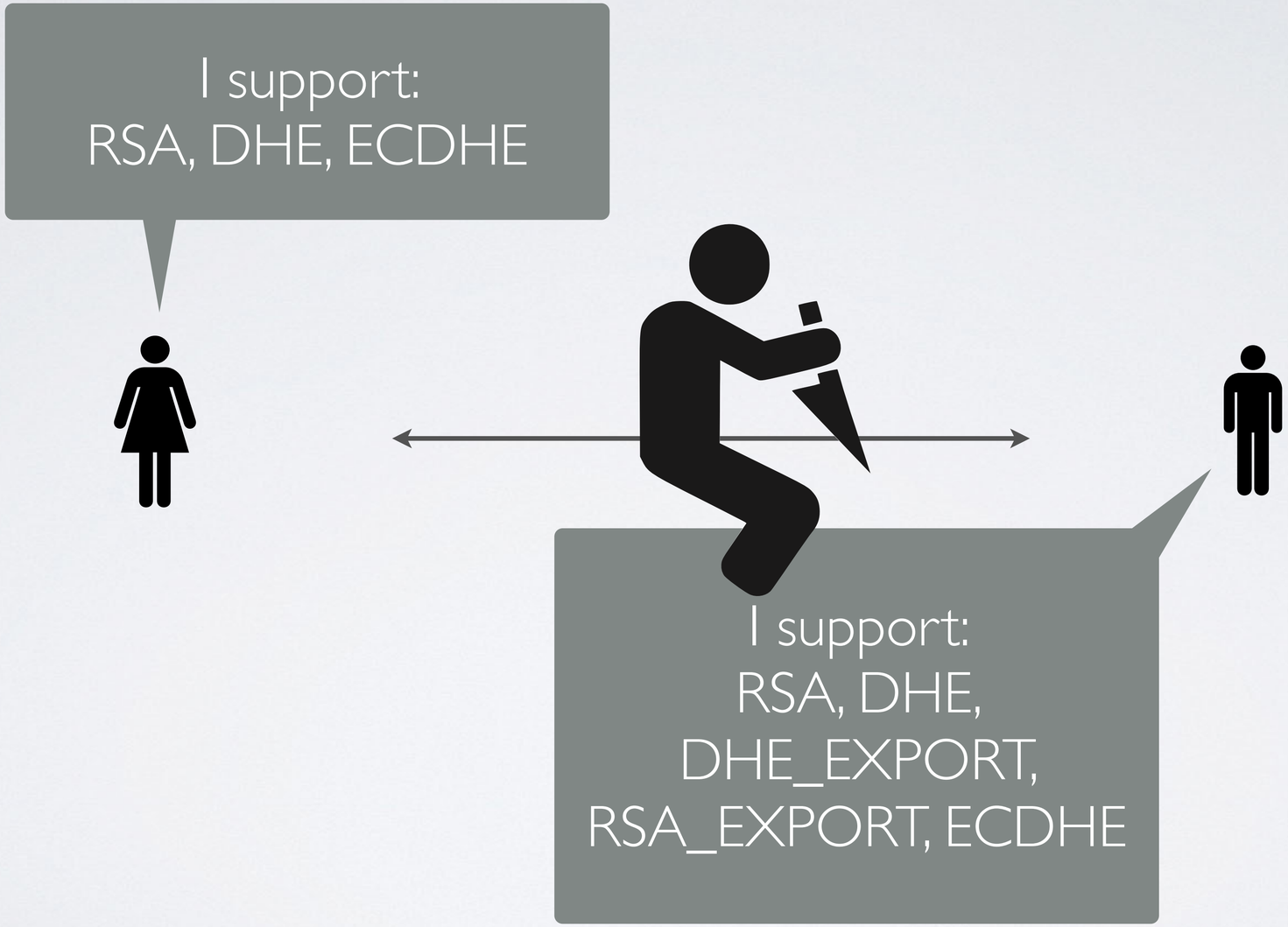
RSA_DHE

Apple issues security patches to protect devices from the FREAK bug

by Mariella Moon | @mariella_moon | 22 days ago

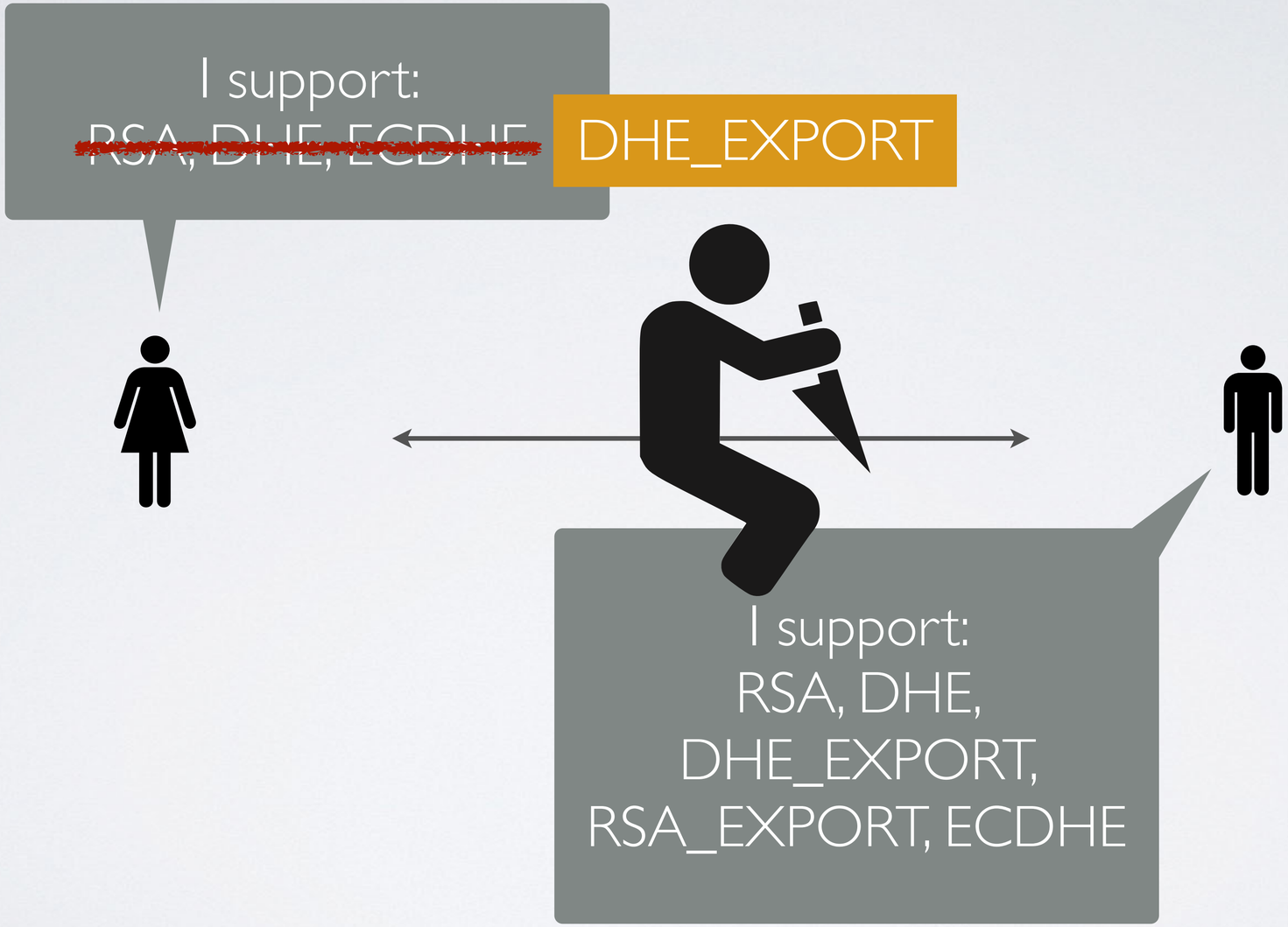
Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

Example 3: Negotiation



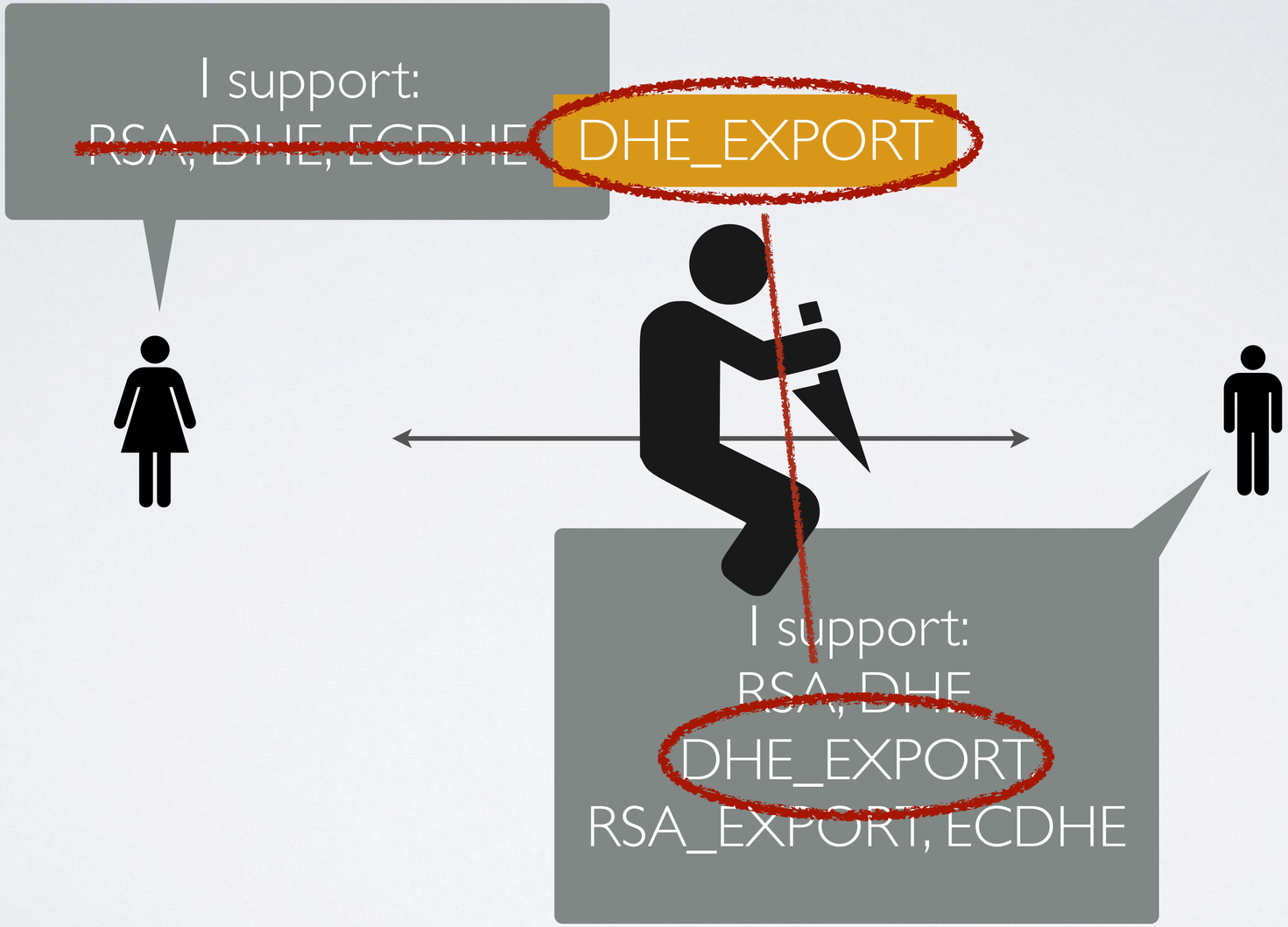
Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

Example 3: Negotiation



Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

Example 3: Negotiation



Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

Example 3: Negotiation

I support:

```
struct {
    select (KeyExchangeAlgorithm) {
        case dh_anon:
            ServerDHParams params;
        case dhe_dss:
        case dhe_rsa:
            ServerDHParams params;
            digitally-signed struct {
                opaque client_random[32];
                opaque server_random[32];
                ServerDHParams params;
            } signed_params;
        case rsa:
        case dh_dss:
        case dh_rsa:
            struct {} ;
           /* message is omitted for rsa, dh_dss, and dh_rsa */
        /* may be extended, e.g., for ECDH -- see [TLSECC] */
    };
} ServerKeyExchange;
```

Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

Example 3: Negotiation

LogJam [Adrian *et al.*]**:**
Due to a bug in *the TLS* ***protocol***
client accepts export-grade DHE key

Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

TLS design/deployment assumptions were wrong, and we knew this for years —
but failed to properly communicate to the community.

TLS design/deployment assumptions were wrong, and we knew this for years —
but failed to properly communicate to the community.

The community made terrible assumptions and didn't ask us what we thought of them. Then they got mired in backwards compatibility issues and only responded to attacks.

Exploiting LogJam

Exploiting LogJam

- To exploit the downgrade attack, requires solving a 512-bit DL in real time
  - Initially this seems challenging, but NFS algorithm can be heavily optimized for pre-computation using only prime (p)
  - "Oversieving" increases cost of sieving and storage, but reduces cost of linear algebra step & final "descent"

Exploiting LogJam

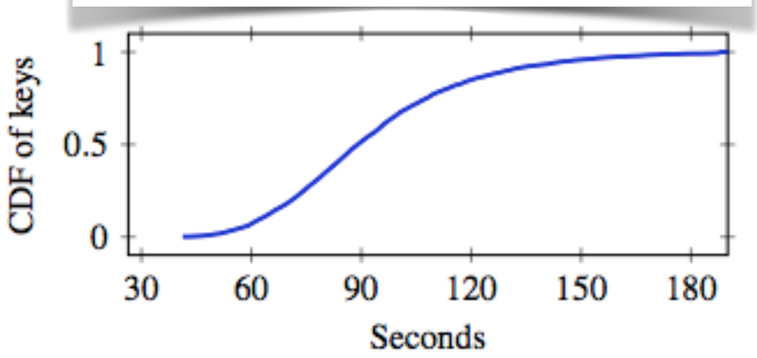
- To exploit the downgrade attack, requires solving a 512-bit DL in real time
  - **92% of DHE_EXPORT servers use one of two hard-coded primes (*p*) (*Mod_SSL, Apache*)**

Exploiting LogJam

- To exploit the downgrade attack, requires solving a 512-bit DL in real time
  - **92% of DHE_EXPORT servers use one of two hard-coded primes (*p*) (*Mod_SSL, Apache*)**

Sieving/Linear Alg:
1 week (wall clock) for each *p*

Descent on (g, h)



Example 3: Negotiation

Short term (hack) solution:

Fix OpenSSL, SecureTransport, SChannel
so they refuse DHE keys <768 bits

patched in NSS, SChannel, BoringSSL, LibreSSL,
SecureTransport

(Took months to accomplish this, since it breaks
~1% of the Internet to make this fix)

RSA_EXPORT, ECDHE

Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

Long(er) term solutions:

Eliminate 1024-bit DHE (but Java).

Stop using common DHE primes.

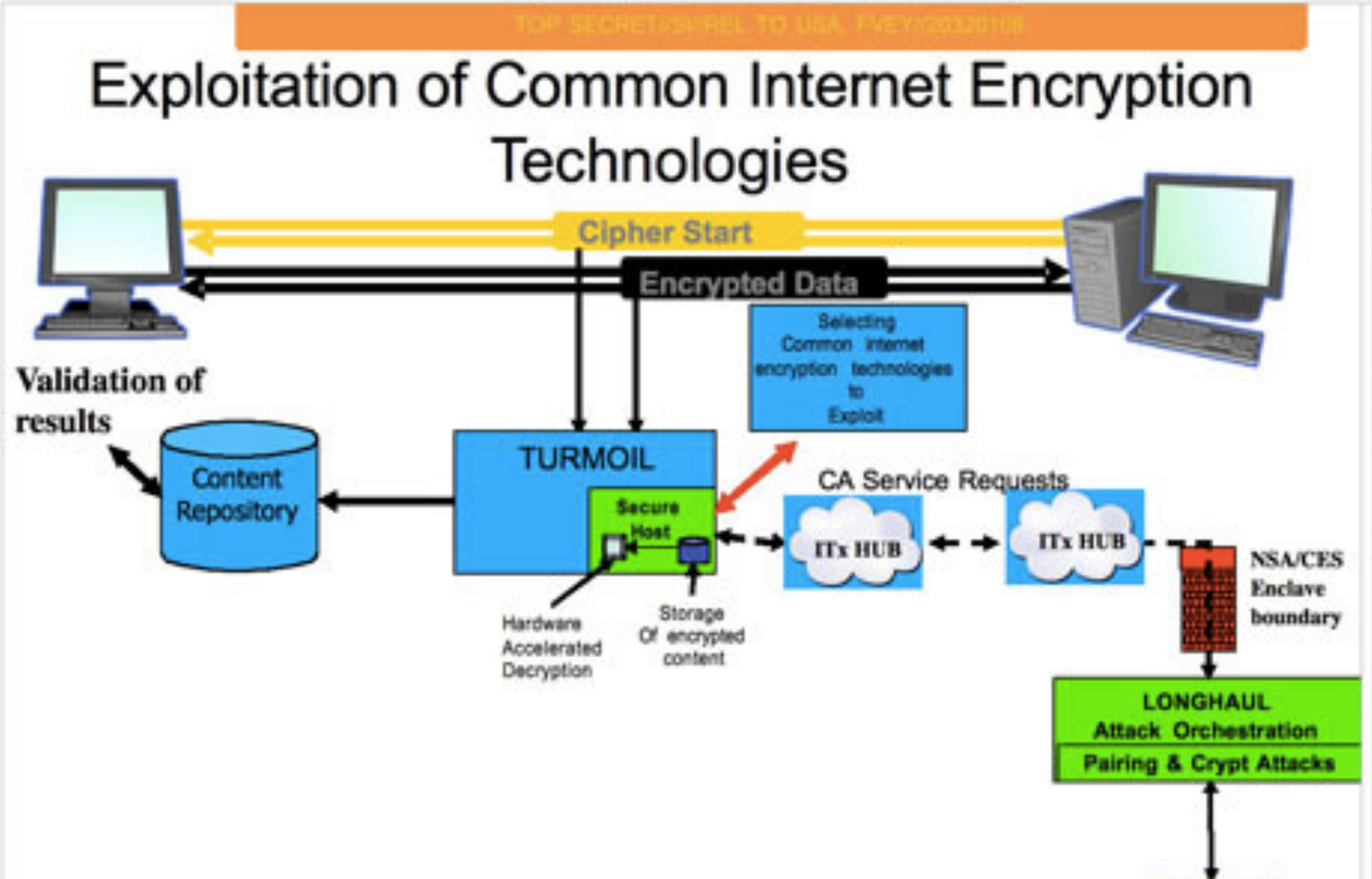
Use EU-CMA signatures to validate the protocol transcript. Then you can achieve the $A \cap B$ security the TLS designers originally set out to achieve.

(TLS 1.3 adds such a message, provisionally.)

Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

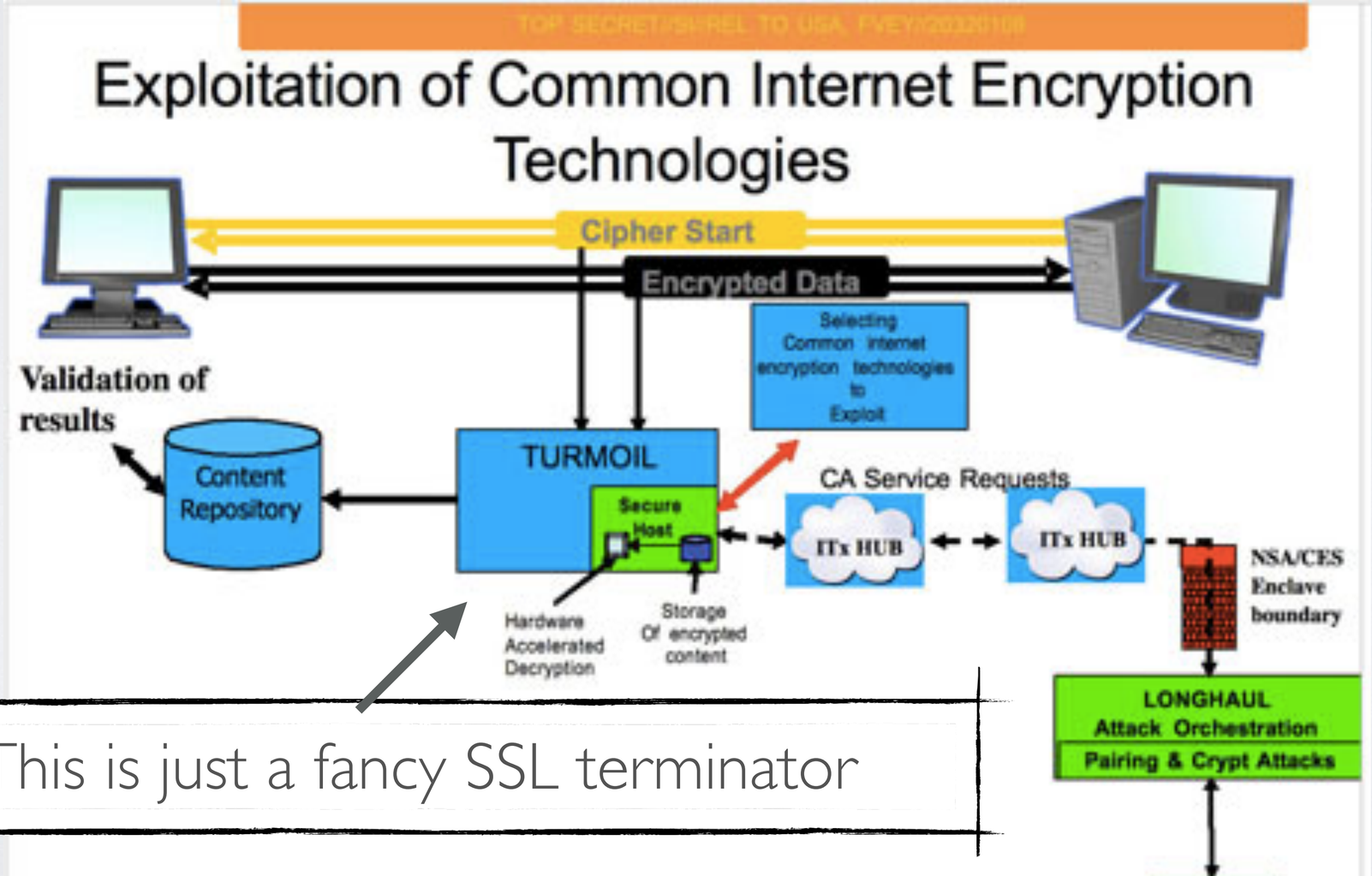
This picture again

- **What's going on here?**



This picture again

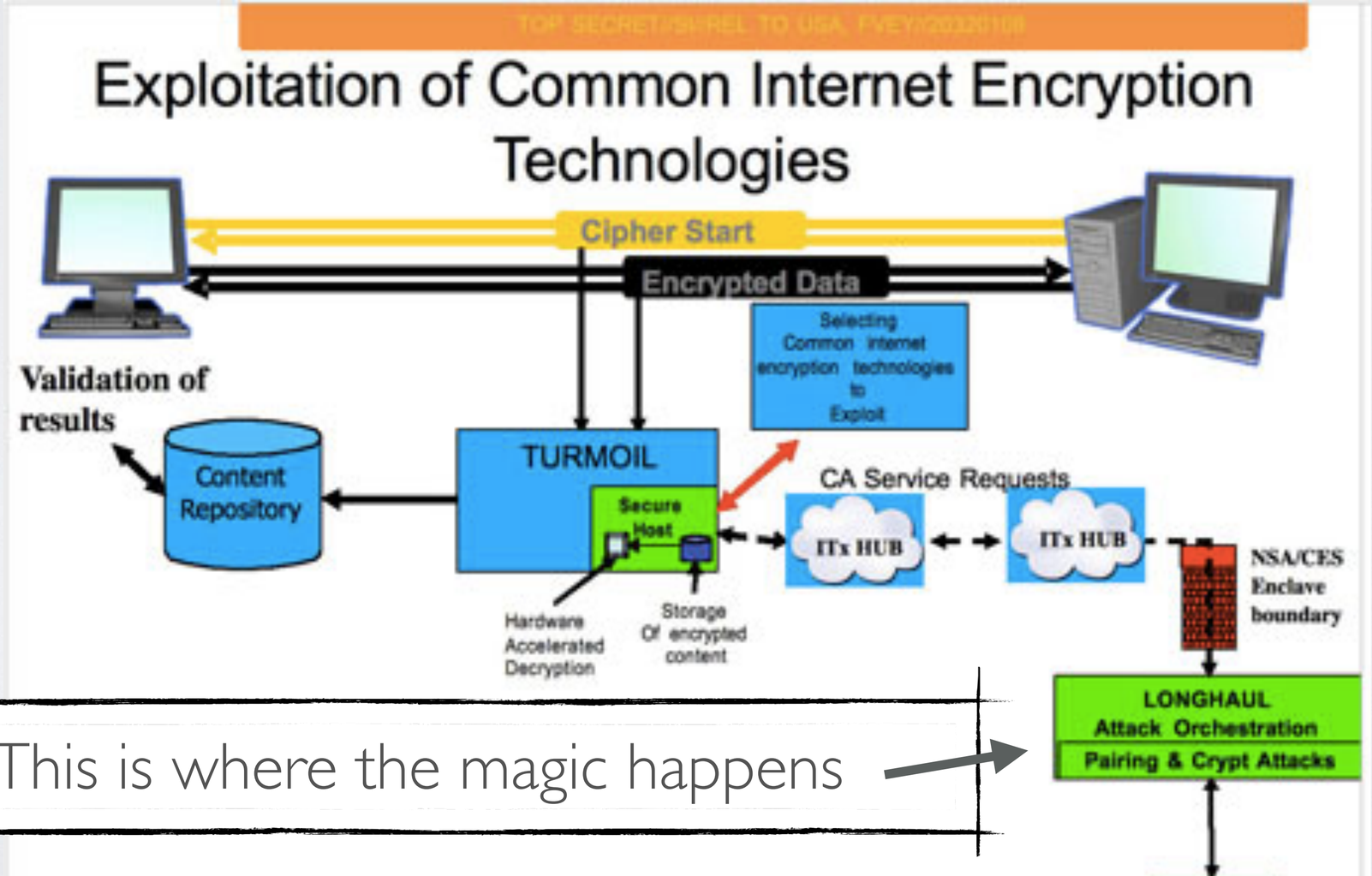
- **What's going on here?**



This is just a fancy SSL terminator

This picture again

- **What's going on here?**



This is where the magic happens

What is LONGHAUL?

(TS//SI/REL) The **LONGHAUL** system provides the **Extended NSA Enterprise** with an end-to-end attack orchestration and key recovery service for Data Network Cipher (DNC) and Data Network Session Cipher (DNSC) traffic. LONGHAUL is extensible to allow for the addition of other Digital Network Intelligence cipher types.

Hypothesis 1: LONGHAUL is a database of stolen RSA secret keys

- This works well, but it's boring
- Easy to solve: switch to PFS cipher suites (DHE/ECDHE)

RSA Exploitation Steps

- Is it the key exchange RSA? (server hello)
  - If so, is the modulus match a known private key? (server certificate)
    - If so, is there 2-sided collect?
      - If so, do we have:
        - Client Hello
        - Server Hello
        - Client Key Exchange

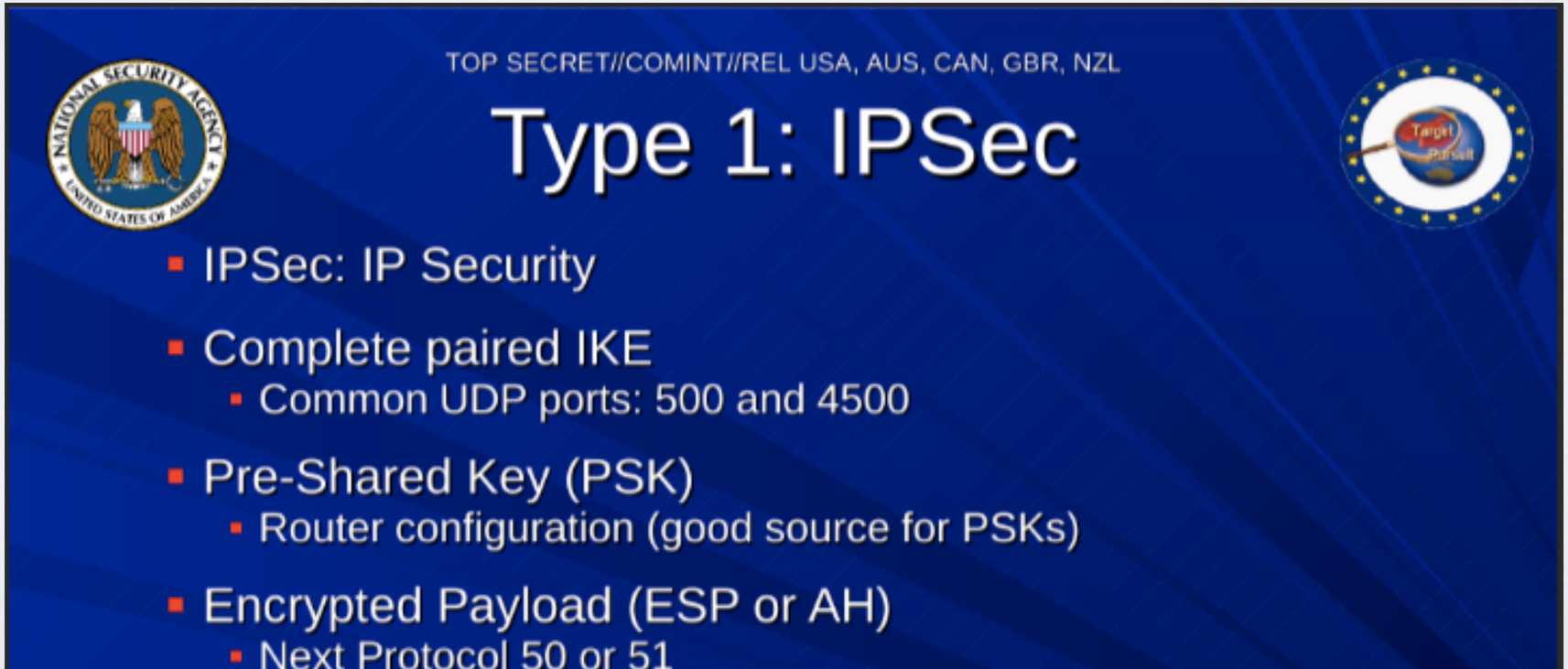
DECRYPTION!

TOP SECRET//COMINT//REL USA, AUS, CAN, GBR, NZL

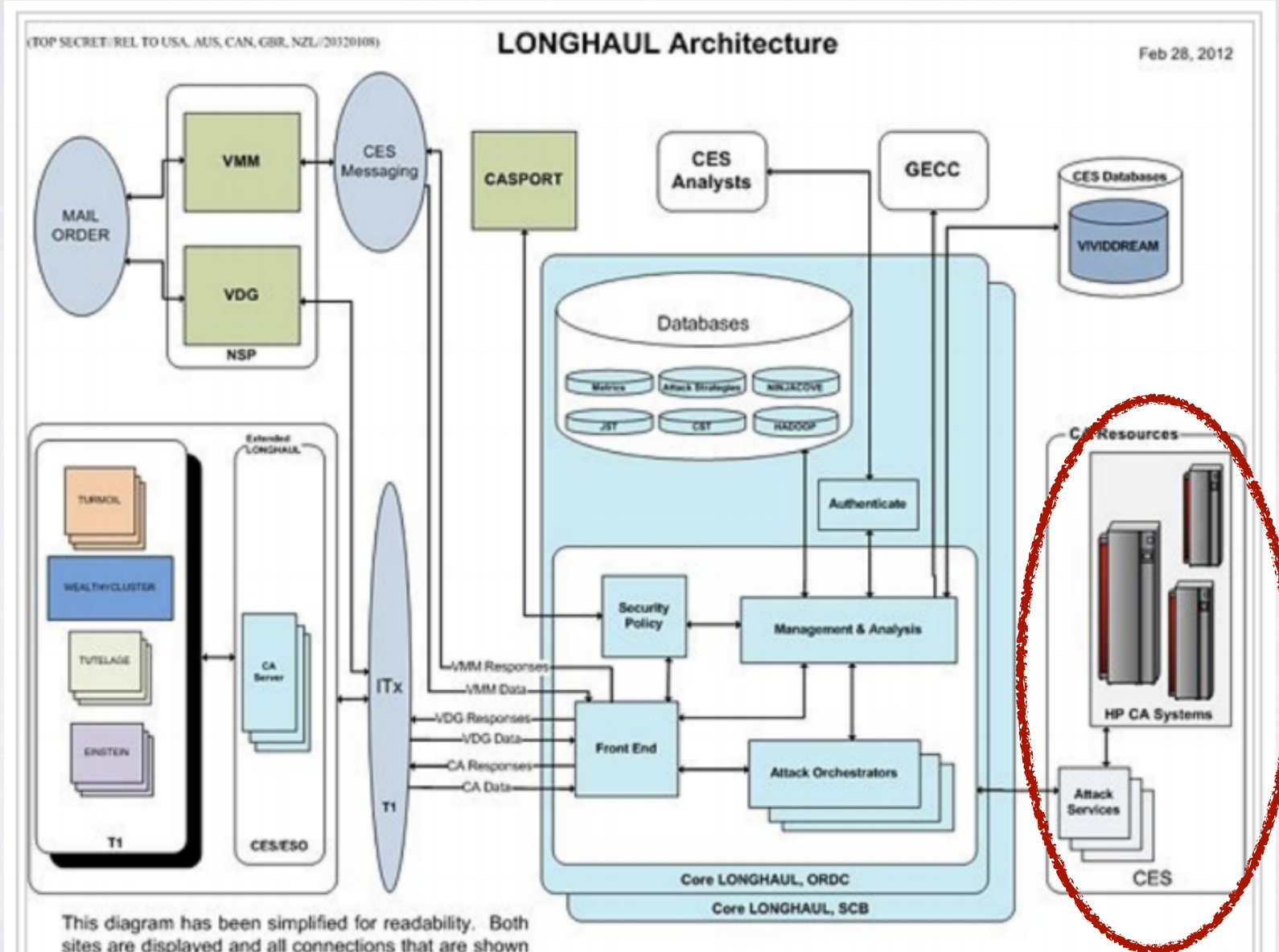
Happy Dance!!

Problem

- LONGHAUL also purports to decrypt IPSec/IKE
  - IKE does not use RSA
  - It uses Diffie-Hellman for each connection.



What is LONGHAUL?



The breakthrough was enormous, says the former official, and soon afterward the agency pulled the shade down tight on the project, even within the intelligence community and Congress. “Only the chairman and vice chairman and the two staff directors of each intelligence committee were told about it,” he says. The reason? “They were thinking that this computing breakthrough was going to give them the ability to crack current public encryption.”

What is LONGHAUL?

(TS//SI/REL) The **LONGHAUL** system provides the **Extended NSA Enterprise** with an end-to-end attack orchestration and key recovery service for Data Network Cipher (DNC) and Data Network Session Cipher (DNSC) traffic. LONGHAUL is extensible to allow for the addition of other Digital Network Intelligence cipher types.

Hypothesis 2: The NSA is breaking1024-bit DHE

- This sounds completely insane
- Maybe it's not

Breaking DHE at scale

- Breaking DHE == solving the Discrete Logarithm problem
  - In theory, this is too expensive for keys >=768 bits
  - However there is a wrinkle…

Breaking DHE at scale

- A large percentage of Apache/Java/ISS servers use *fixed, hardcoded parameters for DHE*
  - IPSec/IKE is even worse: nearly 50% of servers will choose Oakley groups 1 and 2 (768/1024) - generated in 1998
  - NFS is heavily optimized for pre-computation using only the primes
  - **With specific pre-computation ($10s-100s of Million/1 year?)**
    an attacker might be able to break 30-50% of DHE connections with academic levels of computing
  - Approximately 30 core days for final descent

How do we fix *this*?

- Eliminate 1024-bit DH
  - This is challenging in TLS, since many machines (Java 7) crash on longer parameter lengths
  - D. Gillmor, new extension to negotiate FF-DHE
- Eliminate DHE altogether
  - Move to ECDHE, which is currently not 100% supported
  - Downgrade to RSA (!)
- Eliminate common primes

Surely this is all the IETF's fault

Case study: Apple iMessage

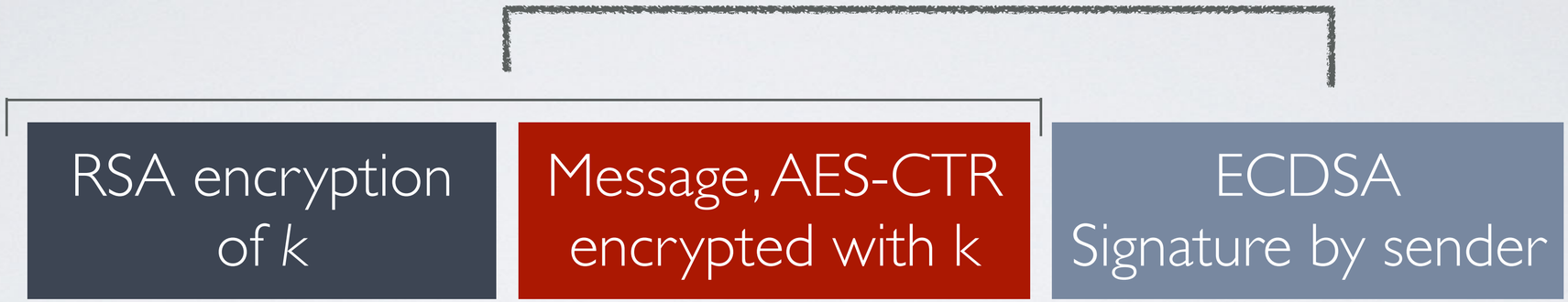
- **<u>Not</u> the most important security protocols on the Internet**
  - But pretty important to real people
  - Once you have messaging, you can build inter-device communications…

iMessage: Encryption

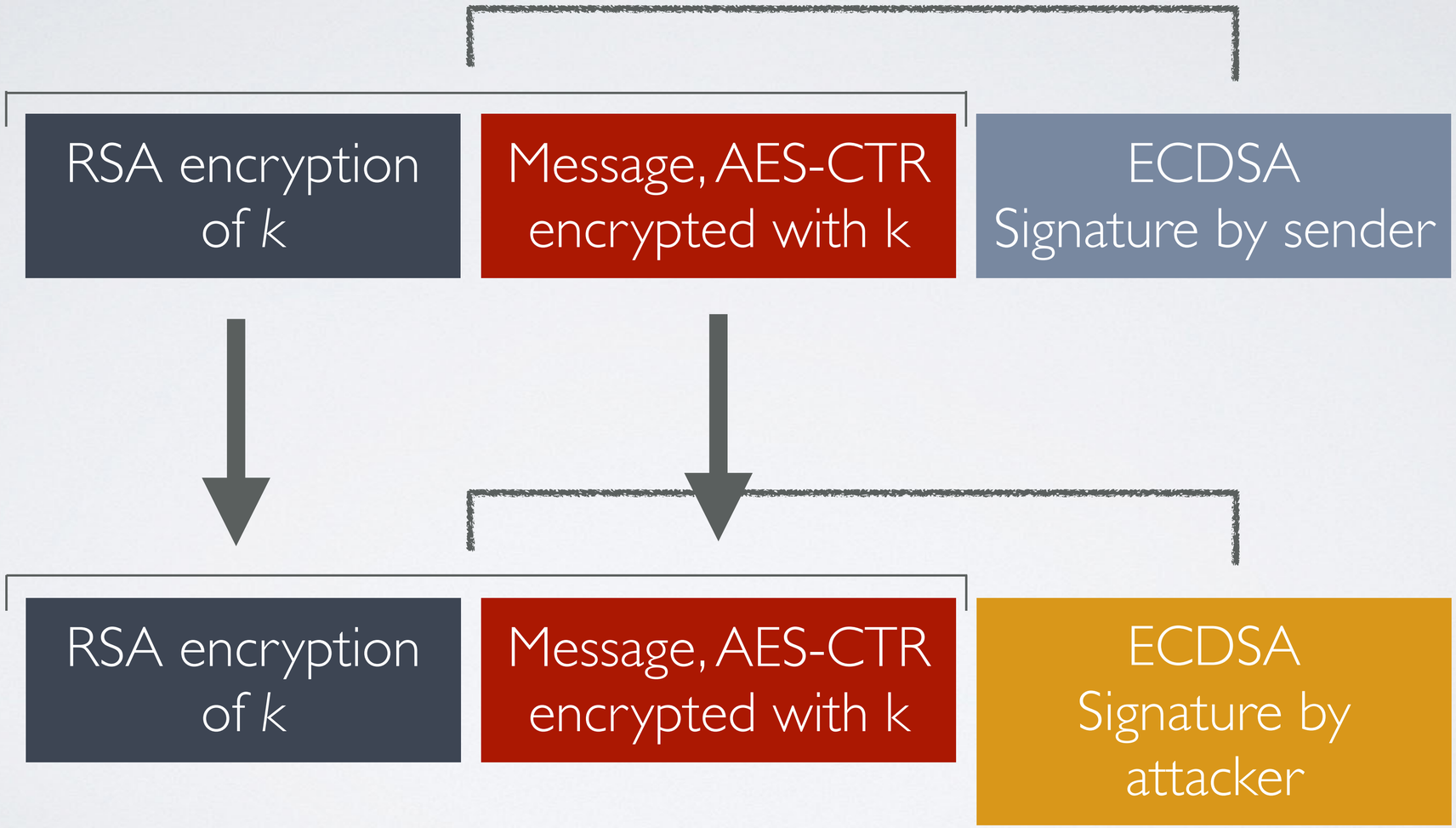
When a user turns on iMessage on a device, the device generates two pairs of keys for use with the service: an RSA 1280-bit key for encryption and an ECDSA 256-bit key on the NIST P-256 curve for signing. The private keys for both key pairs are saved in the device's keychain and the public keys are sent to Apple's directory service (IDS), where they are associated with the user's phone number or email address, along with the device's APNs address.

The user's outgoing message is individually encrypted for each of the receiver's devices. The public RSA encryption keys of the receiving devices are retrieved from IDS. For each receiving device, the sending device generates a random 128-bit key and encrypts the message with it using AES in CTR mode. This per-message AES key is encrypted using RSA-OAEP to the public key of the receiving device. The combination of the encrypted message text and the encrypted message key is then hashed with SHA-1, and the hash is signed with ECDSA using the sending device's private signing key. The

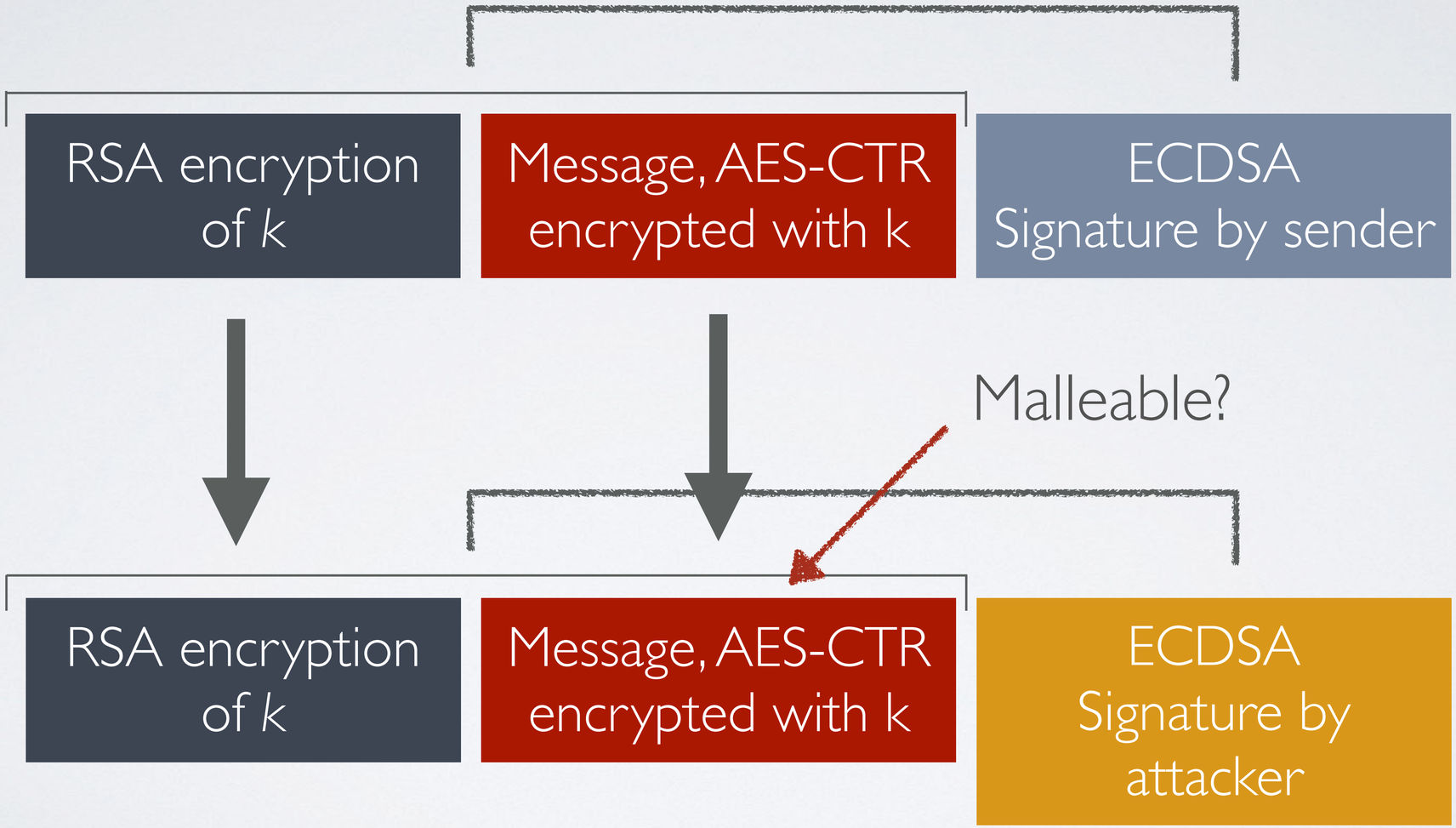
iMessage: Encryption



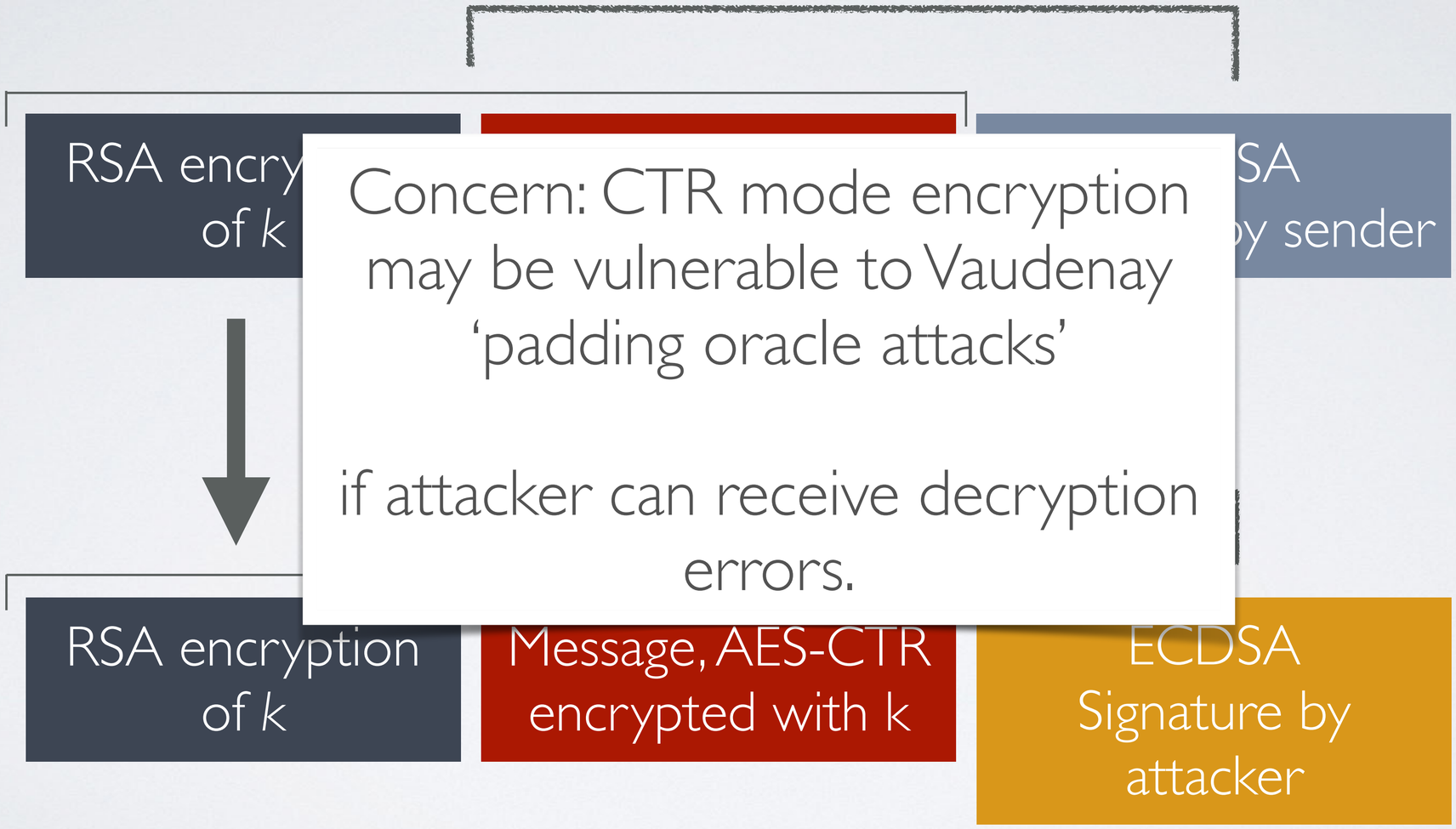
iMessage: Encryption



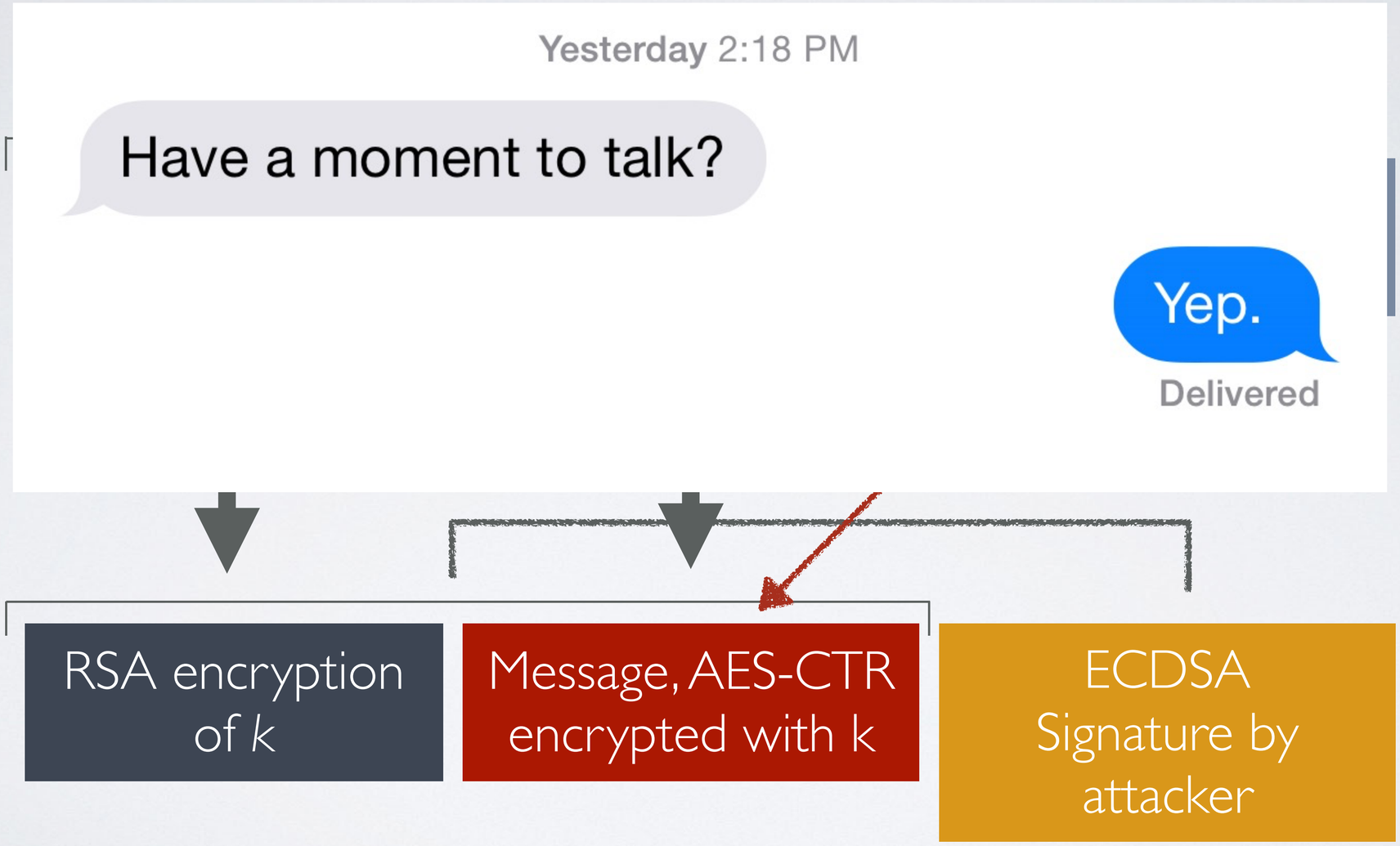
iMessage: Encryption



iMessage: Encryption



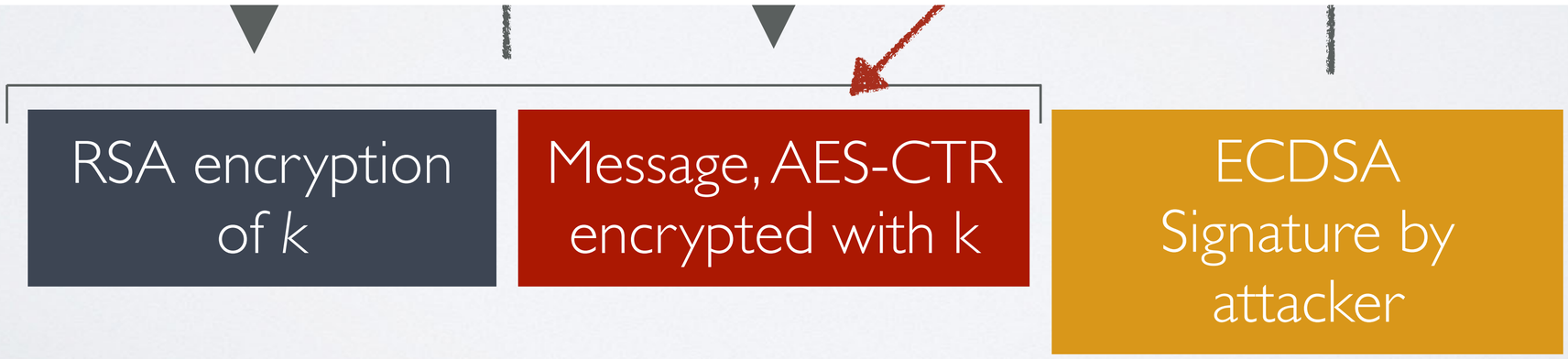
iMessage: Encryption



iMessage: Encryption

Here is an example of such a bplist:

```
D:   True
E:   'pair'
P:   <variable length binary data> (iMessage payload, deflate compressed)
U:   <128bit binary data> (iMessage UID)
c:   100
i:   <32bit integer> (messageId, same as in PUSH header)
sP:  mailto:tim_c@icloud.com  (sender URI)
t:   <256bit binary data> (sender Push-Token)
tP:  mailto:mark_z@facebook.com (receiver URI)
ua:  [Mac OS X,10.8.5,12F37,MacBookPro10,2] (sender OS and hardware version)
v:   1
```



Conclusion

- Cryptography is challenging!
  - We fail to push best practices down to the engineering community
  - They fail to pull best practices from the literature, even years after vulnerabilities are known
  - Cryptosystems continue to become more complex and vulnerable
  - This process is not really tolerable anymore

Why these problems?

- Many problems result from TLS's use of "*pre-historic cryptography*" (- Eric Rescorla)
  - Export grade encryption
  - RSA-PKCS#1v1.5 encryption padding
  - RC4
  - DH parameter generation
  - Horrifying backwards compatibility requirements

Quite a bit

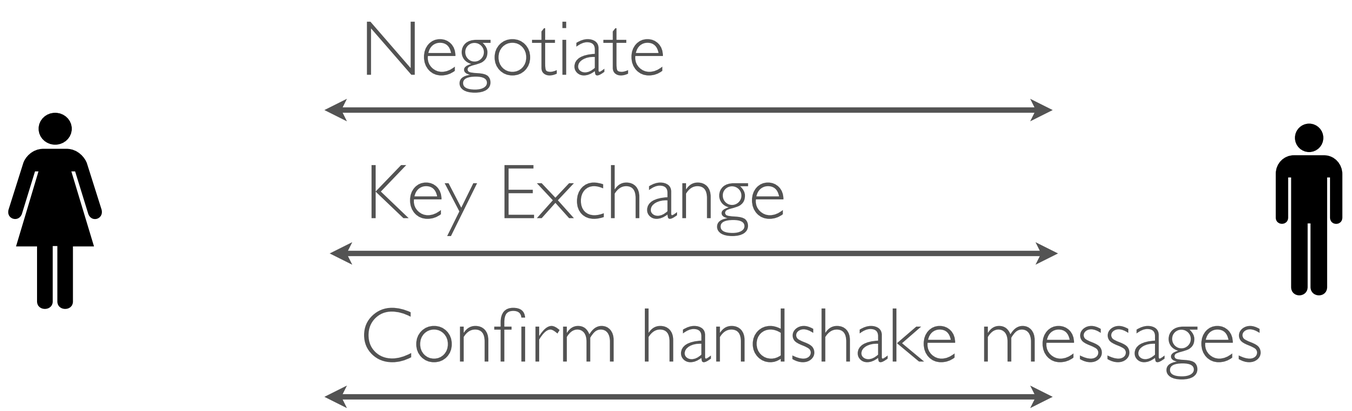
- Many problems result from TLS's use of "*pre-historic cryptography*" (- Eric Rescorla)
  - **Export**
  - RSA-P
  - RC4
  - DH par
  - Horrify

1995-~2000 (and onward)

Weakened "ciphersuites" with limited security
(e.g., 512-bit DH/RSA, 40-bit RC4)

TLS Negotiation

Each TLS handshake begins with a cipher suite negotiation that determines which key agreement protocol (etc.) will be used.



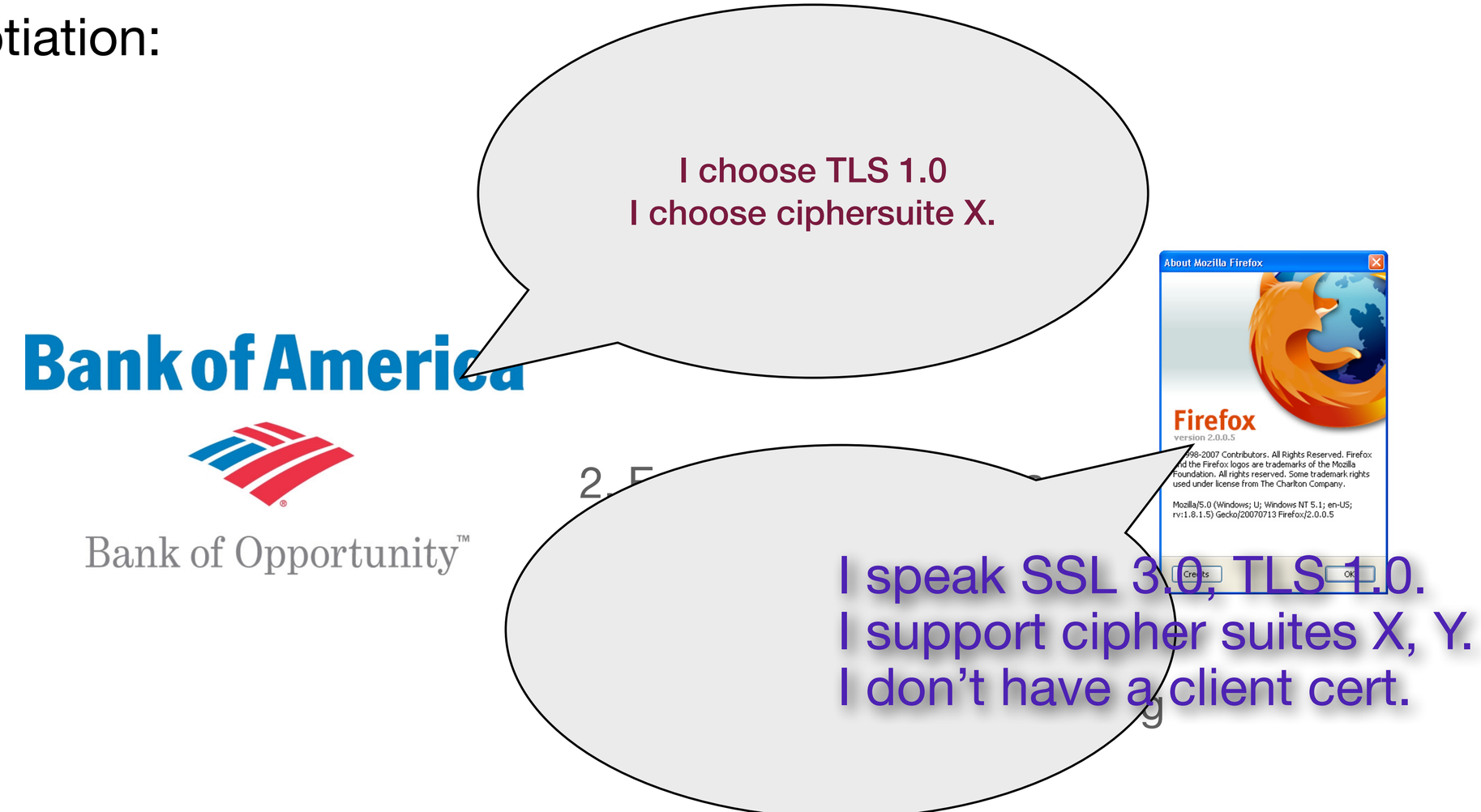
SSL/TLS

Bank of America

Bank of Opportunity™

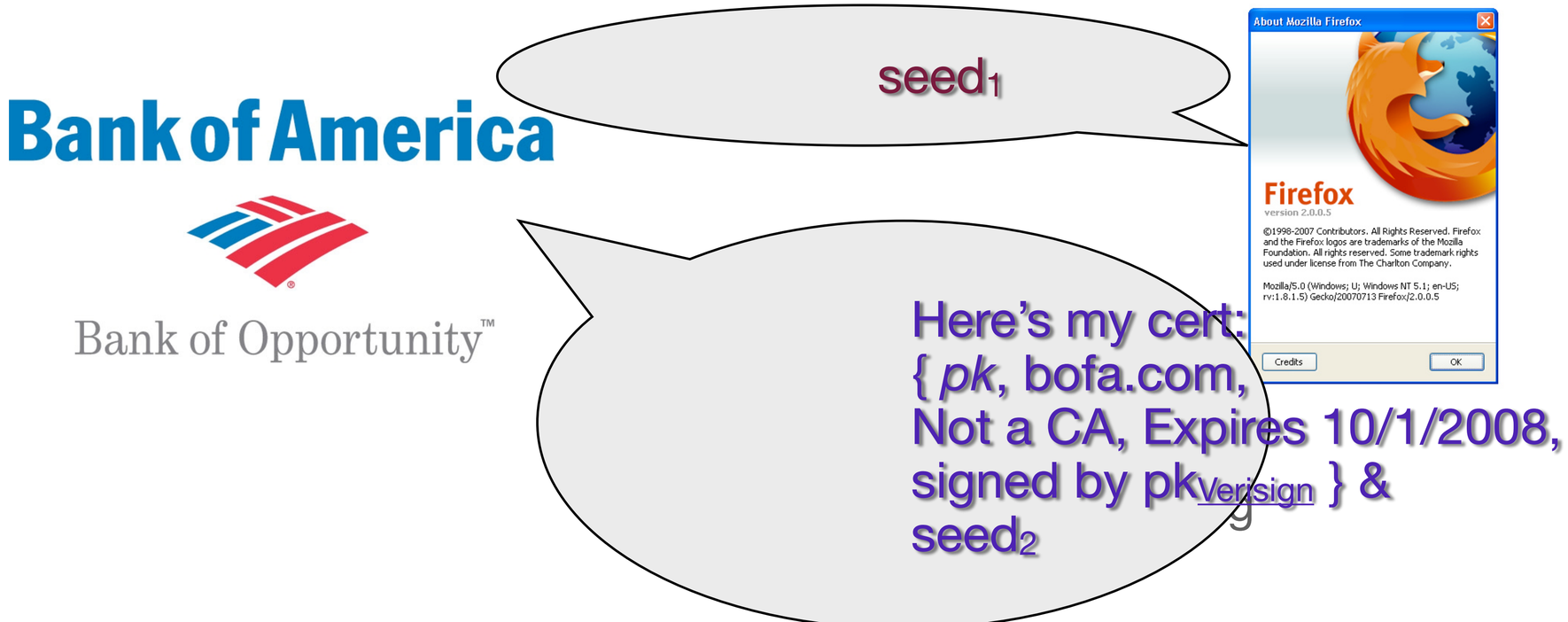
SSL/TLS

- Negotiation:



SSL/TLS

- Certificate Exchange



SSL/TLS

- Session key establishment
  - Various options
  - Common approach: RSA based

$C = RSA\text{-}ENC_{pk}(seed_3)$

1. Negotiate peer capabilities
2. Exchange certificates
3. Secure communi
4. Session expira

$seed_3 = RSA\text{-}DEC(sk, C)$

$k_s = H(seed_1 \,||\, seed_2 \,||\, seed_3)$

$k_s = H(seed_1 \,||\, seed_2 \,||\, seed_3)$

SSL/TLS

- Secure communication
  - In practice, we derive separate MAC & encryption keys

Bank of America

Bank of Opportunity™

1. Negotiate peer capabilities
2. Exchange certificates
3. Session key establishment

5. Session expiration/rekeying

SSL/TLS

- Key expiration/rekeying
  - Key has a defined lifetime
  - If session drops within that lifetime, we restart:

•This shortcut saves PK operations

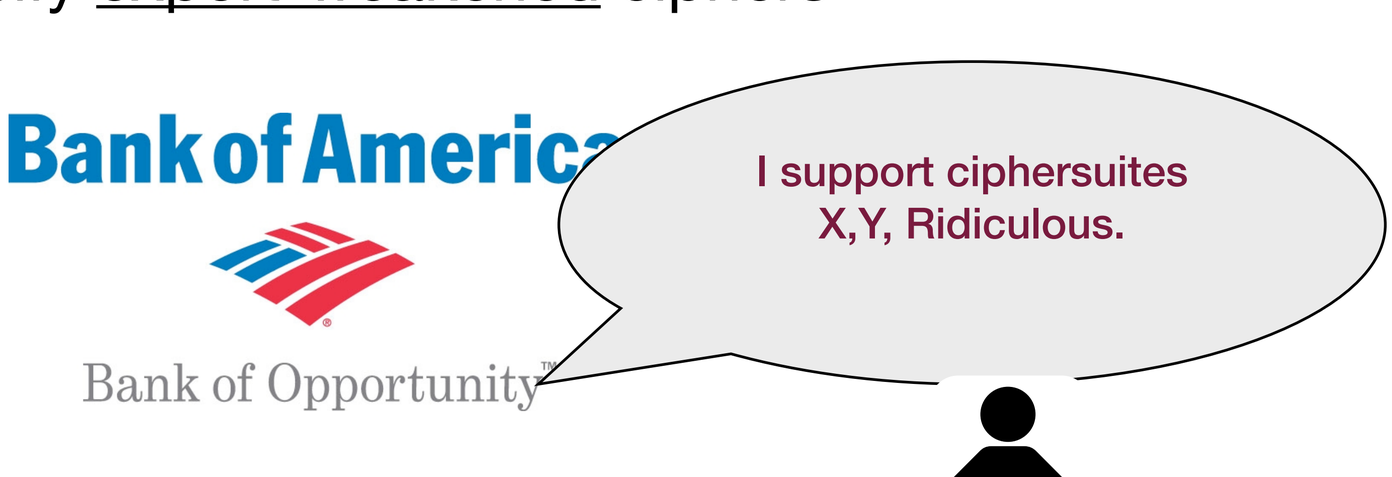
Bank of America

Bank of Opportunity™

1. Negotiate peer capabilities
2. Exchange certificates
3. Session key establishment
4. Secure communications

Attacks on SSL2

- Many and varied...
- Major vulnerability:
  - Ciphersuite list not authenticated
  - Active attacker could modify the message to specify export-weakened ciphers



- Also, uses some non-standard primitives

SSL3

CCS Rollback

- Most messages sent during client/server handshake are authenticated
  - Final MAC is sent at finish message
  - However, [change cipher spec] message is not included in the MAC
  - Tells the other party to start using encryption/authentication
  - Attacker can modify/drop this message!

CCS Rollback

- Normal protocol:

$$
\begin{array}{lll}
\ldots & & \\
1. & C \rightarrow S: & [\textbf{change cipher spec}] \\
2. & C \rightarrow S: & [\textbf{finished:}]\,\{a\}_k \\
3. & S \rightarrow C: & [\textbf{change cipher spec}] \\
4. & S \rightarrow C: & [\textbf{finished:}]\,\{a\}_k \\
5. & C \rightarrow S: & \{m\}_k \\
\ldots & &
\end{array}
$$

CCS Rollback

- MITM attack:

$$
\begin{array}{lll}
\ldots & & \\
1. & C \rightarrow M: & [\textbf{change cipher spec}] \\
2. & C \rightarrow M: & [\textbf{finished:}]\, \{a\}_k \\
2'. & M \rightarrow S: & [\textbf{finished:}]\, a \\
3. & S \rightarrow M: & [\textbf{change cipher spec}] \\
4. & S \rightarrow M: & [\textbf{finished:}]\, \{a\}_k \\
4'. & M \rightarrow C: & [\textbf{finished:}]\, a \\
5. & C \rightarrow M: & \{m\}_k \\
5'. & M \rightarrow S: & m \\
\ldots & &
\end{array}
$$

Key-Exchange Rollback

- SSL3 standard supports two ephemeral key exchange modes:
  - 1. Server publishes ephemeral RSA parameters (signed under its certified signing key)
  - 2. Server publishes ephemeral DH parameters
  - Client may be able to pick which to use
- Why ephemeral key exchange?
  - Advantages of Diffie-Hellman? RSA?

Key Exchange Rollback

Bank of America

Bank of Opportunity™

Normal RSA parameters:
(N,e)

I assume *p* is the RSA modulus, and *g* is the RSA exponent. I ignore the extra value.

Since *p* is a prime, we can compute inverses. Recover k.

Version Rollback

- Release of SSL3 didn’t make SSL2 browsers go away
  - Servers still accepted SSL2 requests
  - Attacker could modify [client hello] message to specify SSL2
  - Server continues with SSL2 connection, attacker uses SSL2 attacks

Version Rollback

- Version rollback is a big problem!
  - SSL, SSH, IPSEC...
  - Example: PPTP

•Can disable encryption, force use of a weaker password authentication protocol

  - Example: L2TP

•Better! But many implementations automatically downgrade to PPTP if L2TP connection fails

Traffic Analysis: SSL3

- Example:
  - First HTTP request typically looks like:

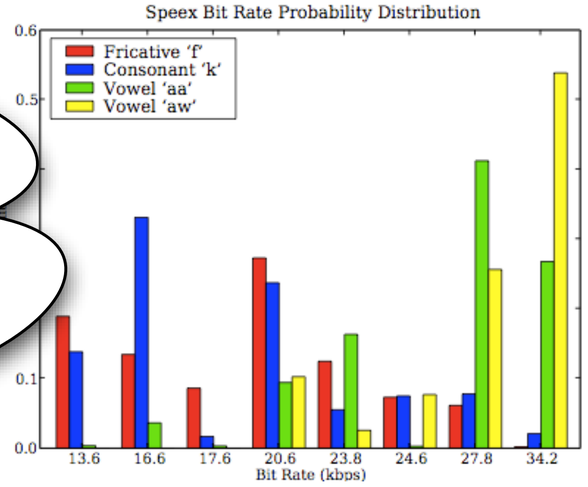
```
/
cnn.com
```

  - From ciphertext length, we may be able to work out URL information

Traffic Analysis++

- Digression: The case of encrypted VoIP
  - Some VoIP protocols use VBR encoding, size of data packets depends on signal
  - Also include “silence suppression” (VAD)
  - Therefore, total traffic is [illegible] related to the contents of

Good news:
Most VoIP implementations don't actually use VBR/ supression

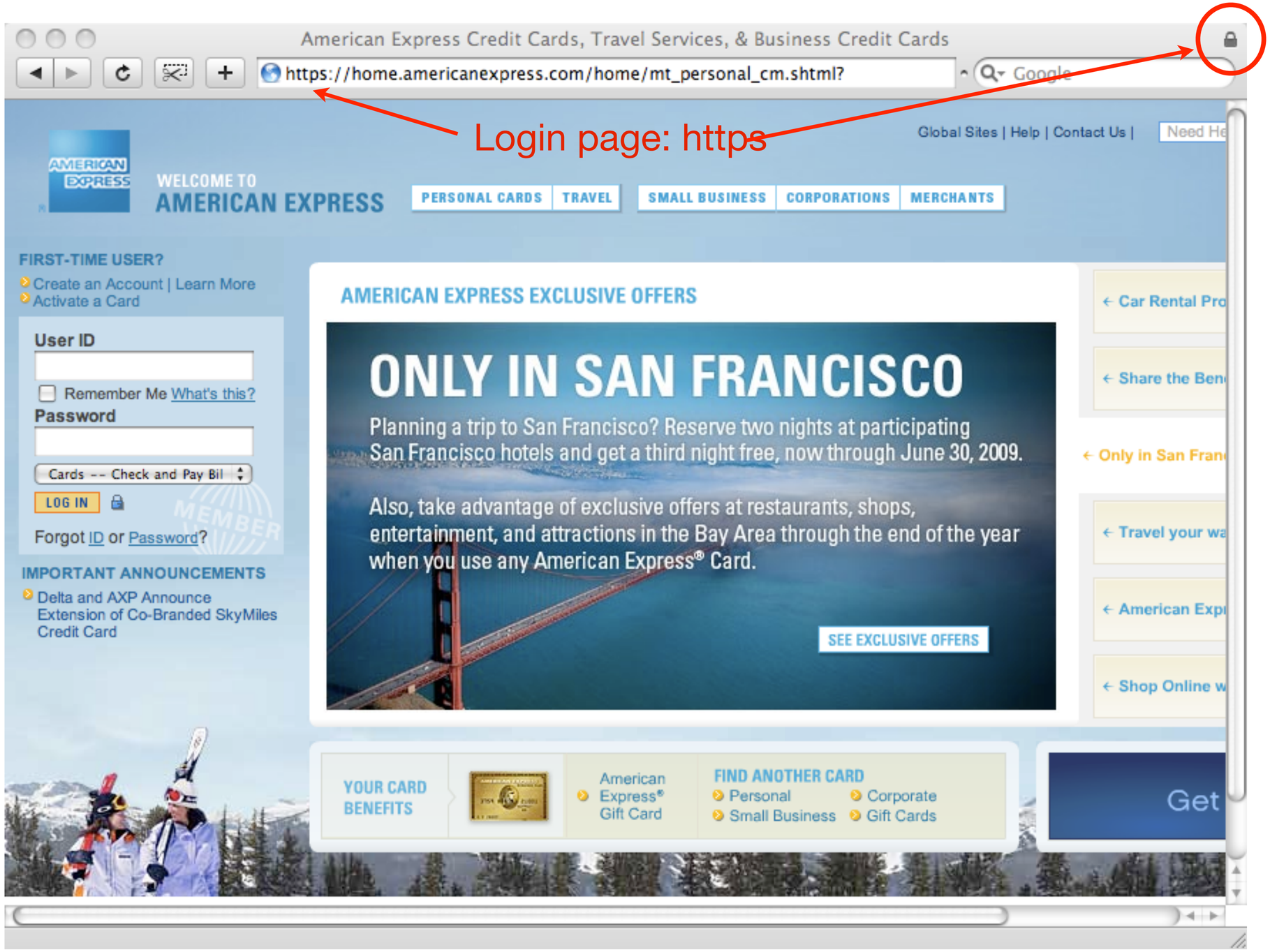


SSL Stripping & Pinning

- Moxie Marlinspike: SSLStrip
- Does not break SSL
- Instead: takes advantage of the way SSL is used

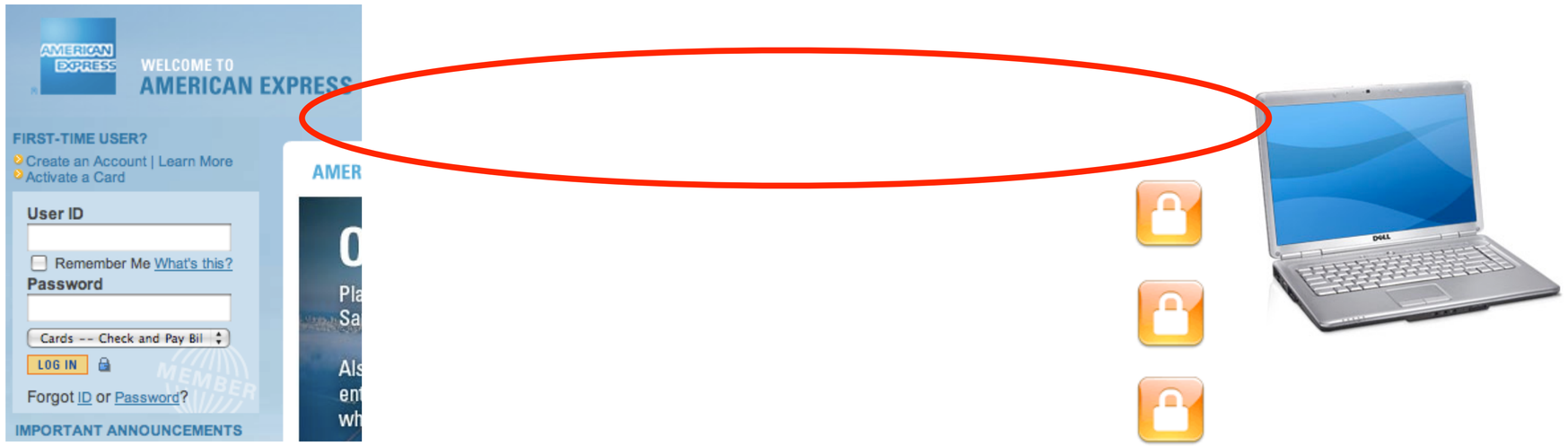
HTTP->HTTPS

- Typical Banking Experience:
  - SSL URLs begin with https://
  - But users rarely type the prefix


American Express Credit Cards, Travel Services, & Business Credit Cards
https://home.americanexpress.com/home/mt_personal_cm.shtml?
Google
Login page: https
Global Sites | Help | Contact Us |
WELCOME TO
AMERICAN EXPRESS
PERSONAL CARDS
TRAVEL
SMALL BUSINESS
CORPORATIONS
MERCHANTS
FIRST-TIME USER?
Create an Account | Learn More
Activate a Card
User ID
Remember Me What's this?
Password
LOG IN
Forgot ID or Password?
IMPORTANT ANNOUNCEMENTS
Delta and AXP Announce Extension of Co-Branded SkyMiles Credit Card
AMERICAN EXPRESS EXCLUSIVE OFFERS
ONLY IN SAN FRANCISCO
Planning a trip to San Francisco? Reserve two nights at participating San Francisco hotels and get a third night free, now through June 30, 2009.
Also, take advantage of exclusive offers at restaurants, shops, entertainment, and attractions in the Bay Area through the end of the year when you use any American Express® Card.
SEE EXCLUSIVE OFFERS
YOUR CARD BENEFITS
American Express® Gift Card
FIND ANOTHER CARD
Personal
Corporate
Small Business
Gift Cards
Get

HTTP->HTTPS

- If you can intercept the user's connection:
  - Don't redirect, or:
  - Redirect to malicious site, unsecured (http)



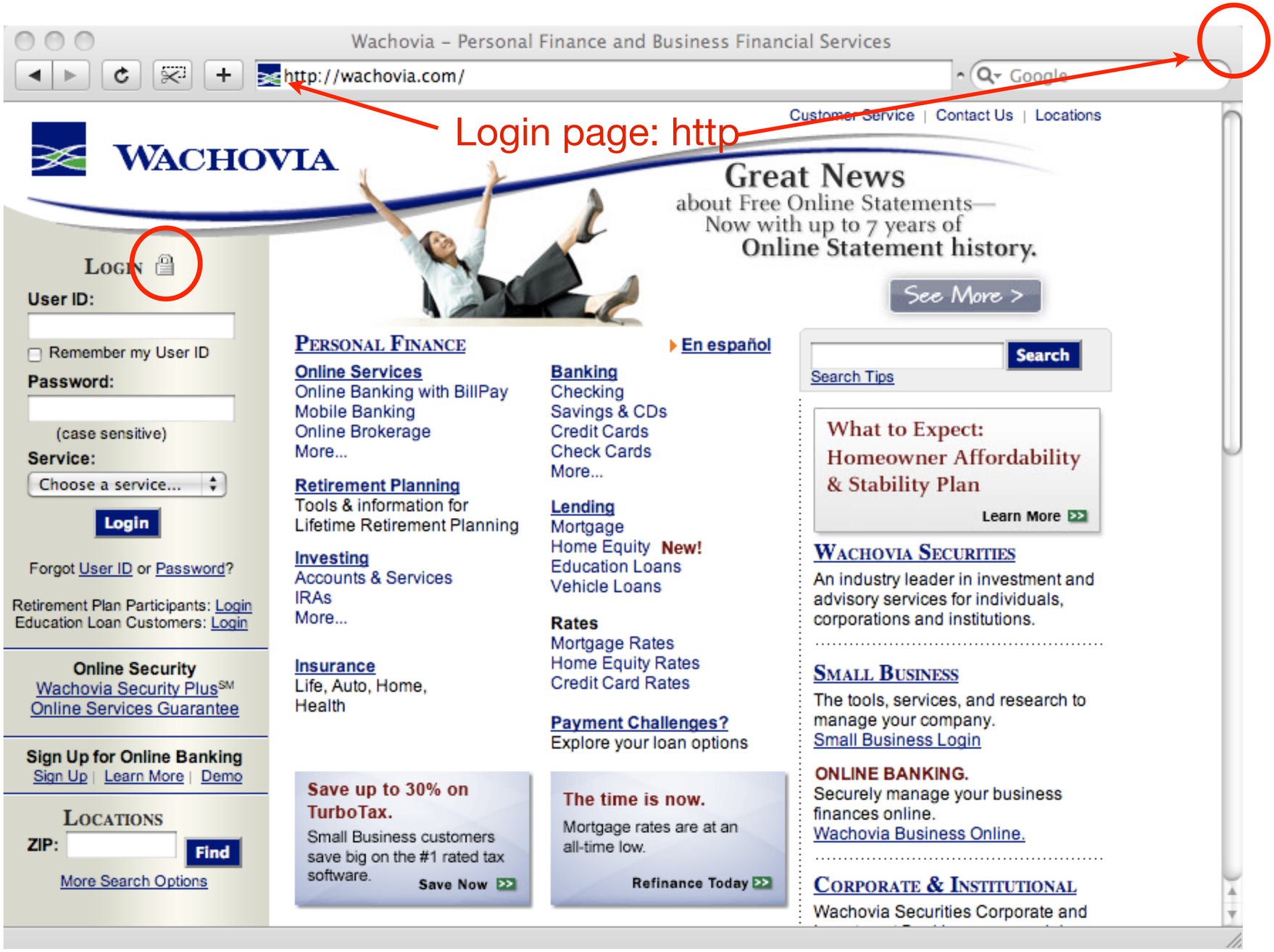
HTTP->HTTPS

- If you can intercept the user's connection:
  - Homograph site: paypaI.com (with a capital i), or:
  - Use clever IDN tricks e.g.,

    https://www.gmail.com/accounts/ServiceLogin!f.ijjk.cn

HTTP->HTTP->HTTPS

- It can be worse:
  - Some sites give an http page with a form that submits via https


Wachovia – Personal Finance and Business Financial Services
http://wachovia.com/
Google
Login page: http
Customer Service | Contact Us | Locations
WACHOVIA
Great News
about Free Online Statements—
Now with up to 7 years of
Online Statement history.
See More >
LOGIN
User ID:
Remember my User ID
Password:
(case sensitive)
Service:
Choose a service...
Login
Forgot User ID or Password?
Retirement Plan Participants: Login
Education Loan Customers: Login
Online Security
Wachovia Security Plus℠
Online Services Guarantee
Sign Up for Online Banking
Sign Up | Learn More | Demo
LOCATIONS
ZIP:
Find
More Search Options
PERSONAL FINANCE
En español
Online Services
Online Banking with BillPay
Mobile Banking
Online Brokerage
More...
Retirement Planning
Tools & information for
Lifetime Retirement Planning
Investing
Accounts & Services
IRAs
More...
Insurance
Life, Auto, Home,
Health
Banking
Checking
Savings & CDs
Credit Cards
Check Cards
More...
Lending
Mortgage
Home Equity New!
Education Loans
Vehicle Loans
Rates
Mortgage Rates
Home Equity Rates
Credit Card Rates
Payment Challenges?
Explore your loan options
Save up to 30% on TurboTax.
Small Business customers save big on the #1 rated tax software.
Save Now
The time is now.
Mortgage rates are at an all-time low.
Refinance Today
Search
Search Tips
What to Expect:
Homeowner Affordability & Stability Plan
Learn More
WACHOVIA SECURITIES
An industry leader in investment and advisory services for individuals, corporations and institutions.
SMALL BUSINESS
The tools, services, and research to manage your company.
Small Business Login
ONLINE BANKING.
Securely manage your business finances online.
Wachovia Business Online.
CORPORATE & INSTITUTIONAL
Wachovia Securities Corporate and

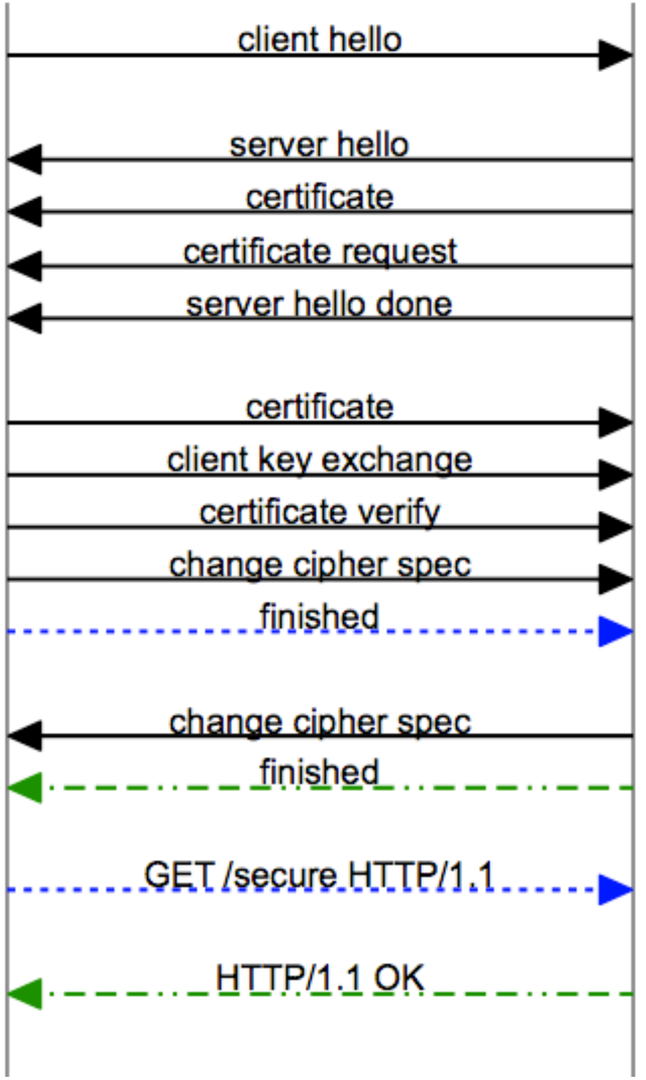
Injecting Prefixes

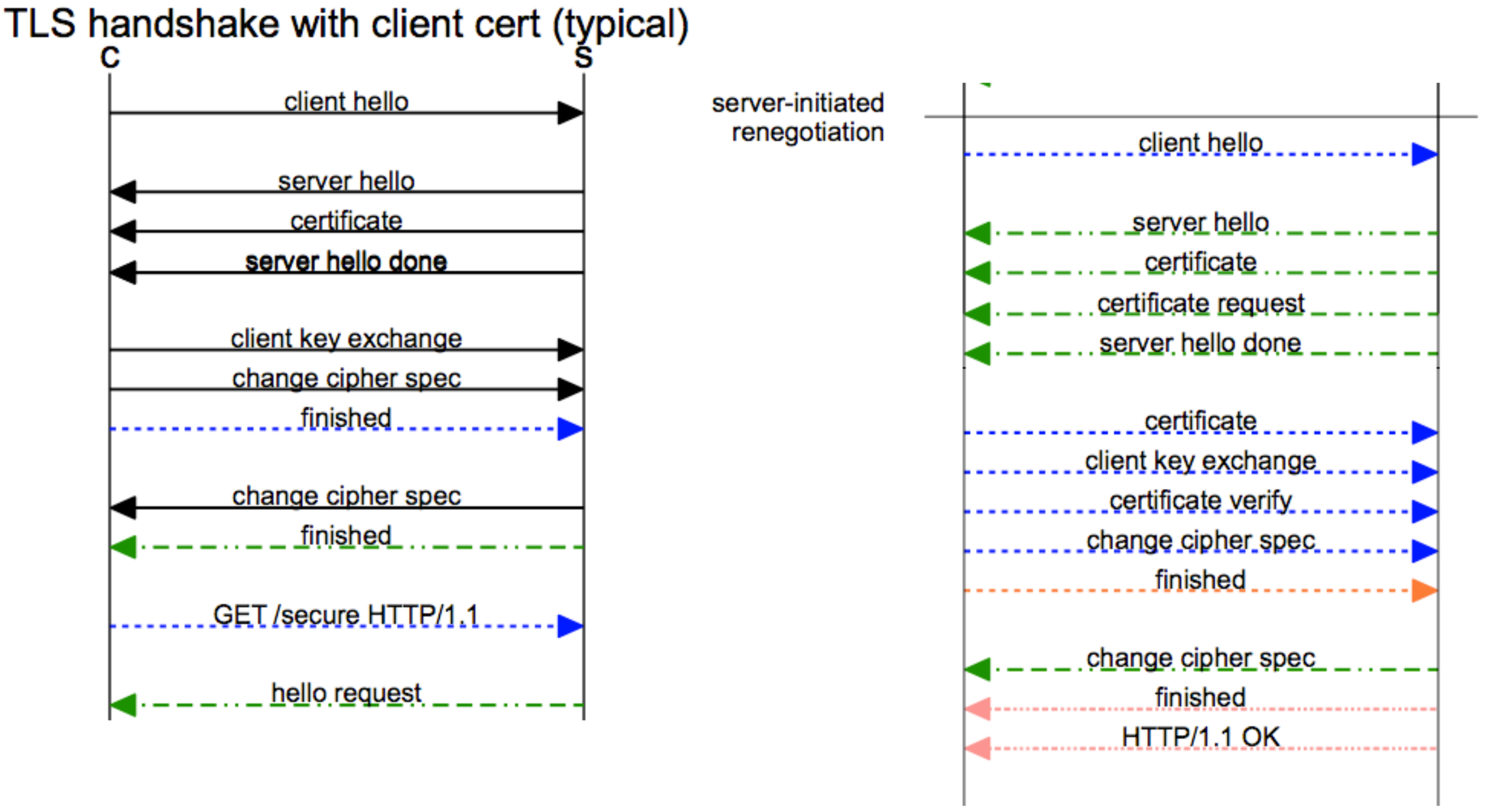
- Ray, Dispensa, 11/2009:
  - Many web servers require client-side auth, but only for certain resources

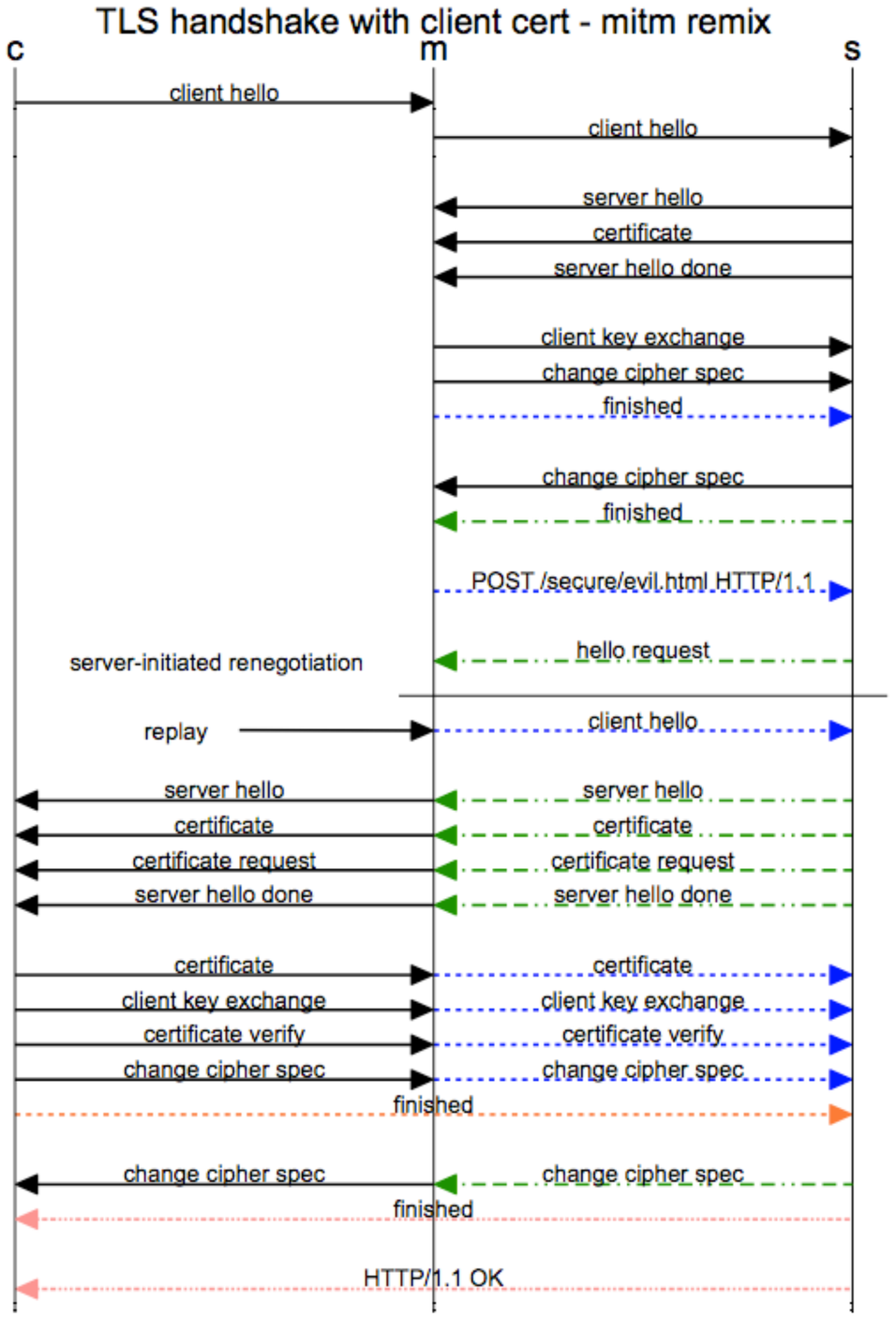
```
GET /highsecurity/index.html HTTP/1.1
Host: example.com
Connection: keep-alive
```

  - This may require an on-the-fly TLS re-negotiation

TLS handshake with client cert (ideal)




TLS handshake with client cert (typical)
C
S
client hello
server hello
certificate
server hello done
client key exchange
change cipher spec
finished
change cipher spec
finished
GET /secure HTTP/1.1
hello request
server-initiated
renegotiation
client hello
server hello
certificate
certificate request
server hello done
certificate
client key exchange
certificate verify
change cipher spec
finished
change cipher spec
finished
HTTP/1.1 OK


TLS handshake with client cert - mitm remix
c
m
s
client hello
client hello
server hello
certificate
server hello done
client key exchange
change cipher spec
finished
change cipher spec
finished
POST /secure/evil.html HTTP/1.1
server-initiated renegotiation
hello request
replay
client hello
server hello
certificate
certificate request
server hello done
server hello
certificate
certificate request
server hello done
certificate
client key exchange
certificate verify
change cipher spec
finished
certificate
client key exchange
certificate verify
change cipher spec
change cipher spec
finished
HTTP/1.1 OK

DECT

- Digital Enhanced Cordless Telephone protocol
  - European standard, now in US
  - Interoperable devices
  - Connects base station (FT) to handsets (PT)
- Tools:
  - DECT Standard Cipher (DSC)
  - DECT Standard Authentication Algorithm (DSAA)

DECT

- Step 1: Pairing
  - User enters a 4-digit PIN into handset and base
  - Base generates a 64-bit seed, combined with PIN to generate shared key (UAK)
  - Base and handset conduct challenge/re

Total entropy of UAK:
77.288 bits (64-bit seed + PIN)
Much less if PRNG is bad!

DECT

- Step 2: Authentication
  - Two … commended one:

In common mode, <u>only the handset</u> is authenticated!

DECT Attack

- Step 2: Authentication
  - Two commended one:

ode?

DECT, other

- A11, A12 built from weak cipher
  - Authors show how this cipher can be inverted using some clever attacks
  - Leaves room for attacks
    even if protocol bug fixed
  - Eerily reminiscent of GSM...

•Weak protocols

•Weak homebrew ciphers

Example: DTCP

- BluRay & HD-DVD Disks
  - Contains “protected” area that can’t be read using normal Drive protocol
  - Embeds secret “Binding Nonce”
- 

DTCP Protocol

- Digital Transmission Content Protection
  - Runs between Drive and Host
  - Encrypts & Authenticates Communications

DTCP

- One layer of protection for HD-DVD/BluRay
  - Encrypts/authenticates content traversing unprotected bus lines

Attacker

DTCP AKE

- Authenticated Key Exchange
  - EC Diffie-Hellman Protocol
  - Each device has a certificate & secret key
  - Devices also have a certificate revocation list, to prevent communication with hacked devices

DTCP AKE (v1.4)

$r_A, Cert_A$

$r_B, Cert_B$

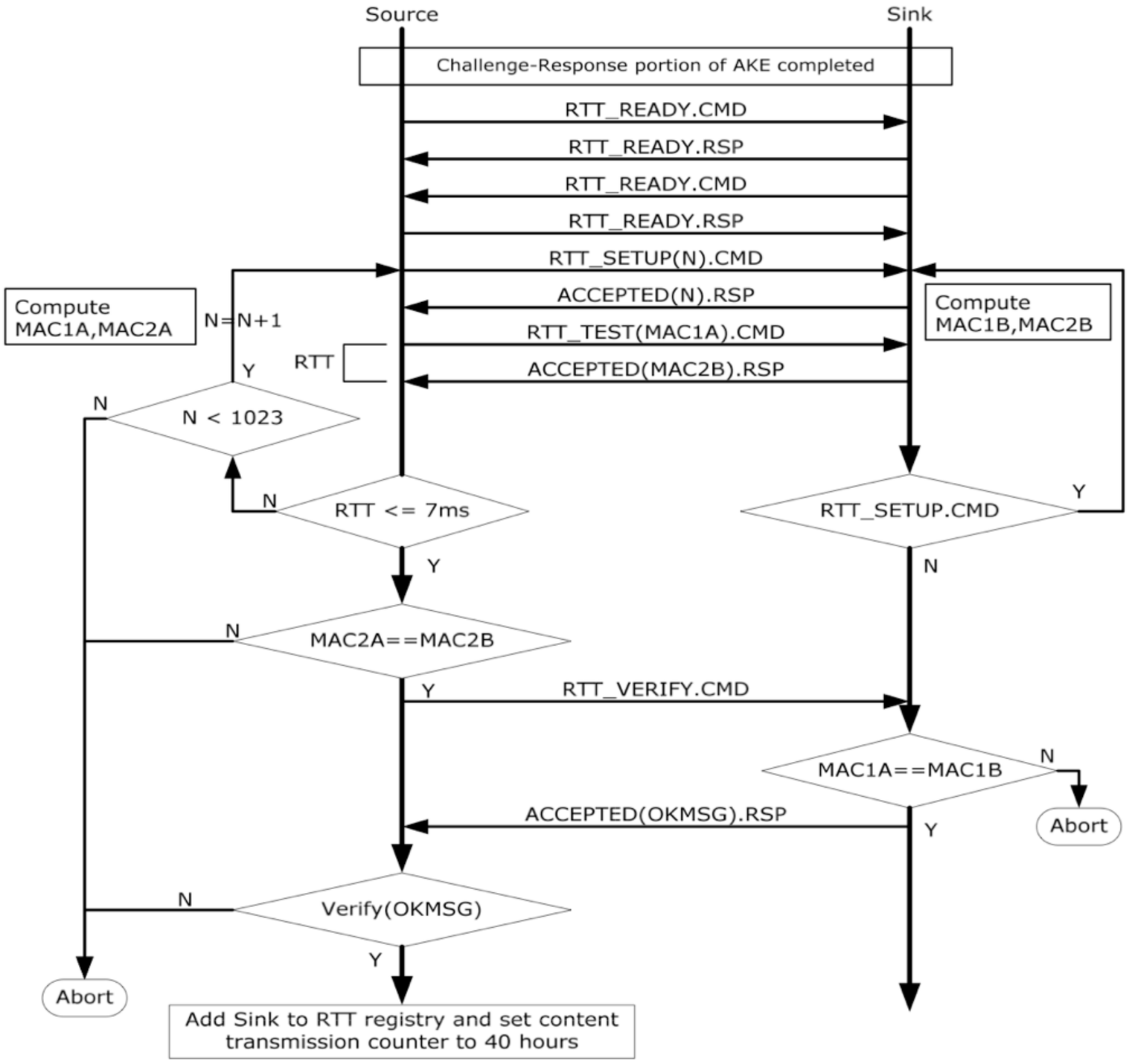
$g^a, Text_A, Sign_{sk_A}(r_A, g^a, Text_A)$

$g^b, Text_B, Sign_{sk_B}(r_B, g^b, Text_B)$

Compute shared key

Verify Shared Key (Localization)

Talk!


Source
Sink
Challenge-Response portion of AKE completed
RTT_READY.CMD
RTT_READY.RSP
RTT_READY.CMD
RTT_READY.RSP
RTT_SETUP(N).CMD
ACCEPTED(N).RSP
RTT_TEST(MAC1A).CMD
ACCEPTED(MAC2B).RSP
Compute MAC1A,MAC2A
N=N+1
RTT
Compute MAC1B,MAC2B
Y
N
N < 1023
N
RTT <= 7ms
RTT_SETUP.CMD
Y
Y
N
N
MAC2A==MAC2B
Y
RTT_VERIFY.CMD
MAC1A==MAC1B
N
Abort
ACCEPTED(OKMSG).RSP
Y
N
Verify(OKMSG)
Y
Abort
Add Sink to RTT registry and set content transmission counter to 40 hours

Other Attacks

- Replay Attacks
  - Attacker replays older messages
  - Can be countered with timestamps, nonces and sequence counters
- Cut & Paste
  - Malleable encryption scheme like CBC
  - Can be countered with MACs
- Reflection
  - If party A sends a message, just bounce it back

Discussion

- We’ve seen standards with problems
  - Usually the cryptanalysis comes after the standard is released, and products in the field
  - Why?

Next Time

- Next lecture:
  - How do we design secure protocols?

END

Ciphersuite Rollback

Bank of America

Bank of Opportunity™

Ciphersuite Rollback

Bank of America

Bank of Opportunity™

Ciphersuite Rollback

- Big caveat:
  - Only works when client asks for authentication <u>without</u> encryption

Bank of America

Bank of Opportunity™

Server thinks encryption is disabled, but gets an encrypted MAC