Secure Protocols in a Hostile World

for CHES 2015

Matthew Green
Johns Hopkins University

Why this talk?

10 Things You Should Know
About Computer Security

5: Cryptography is a Solved Problem

Cryptography: The strongest link in the chain*

but not to others. Unfortunately, people concentrate too much on the cryptography of a system – which is the equivalent of strengthening the strongest link in a chain.

Why this presentation?

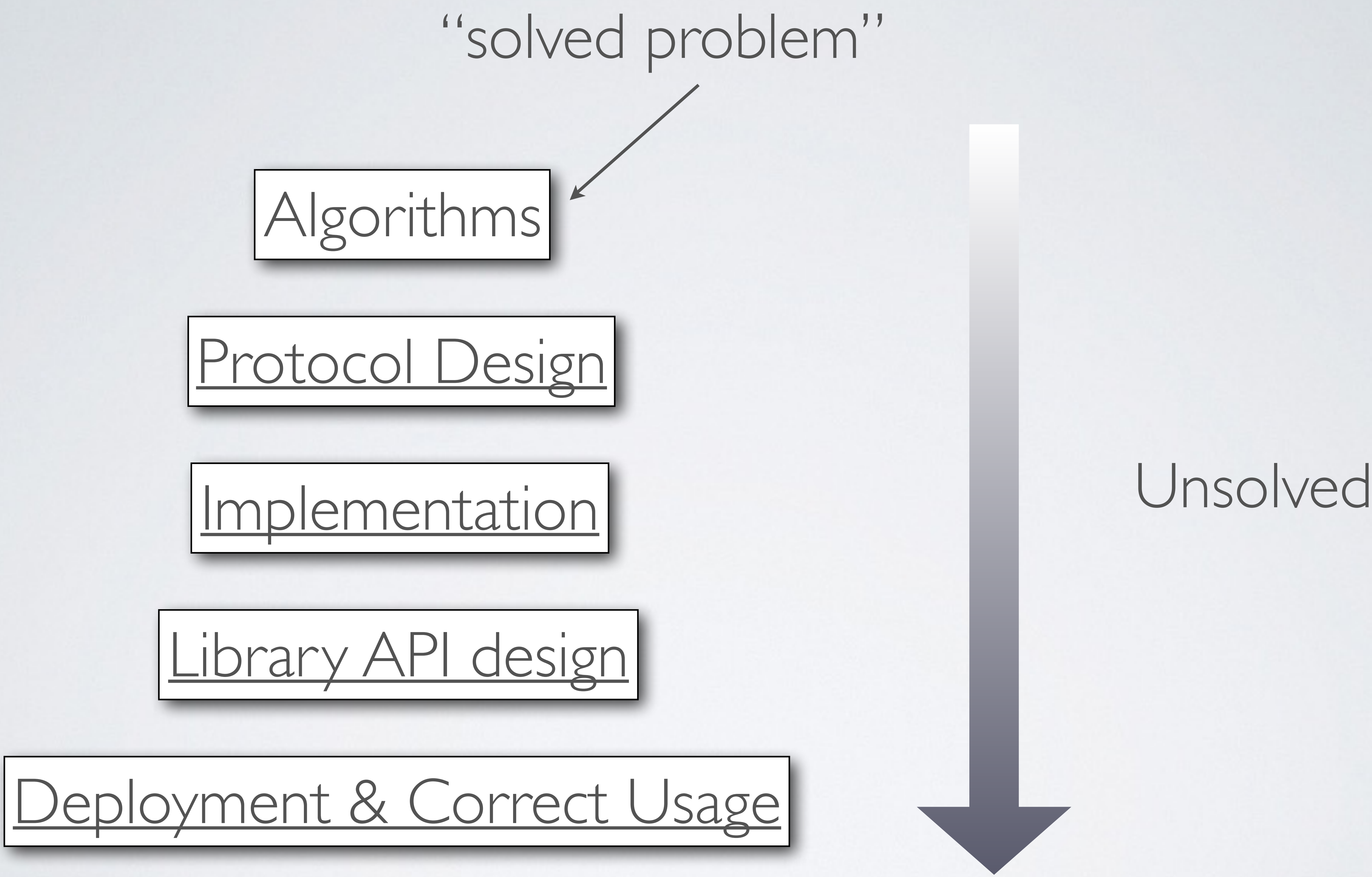
ings You Should Know
mputer Security

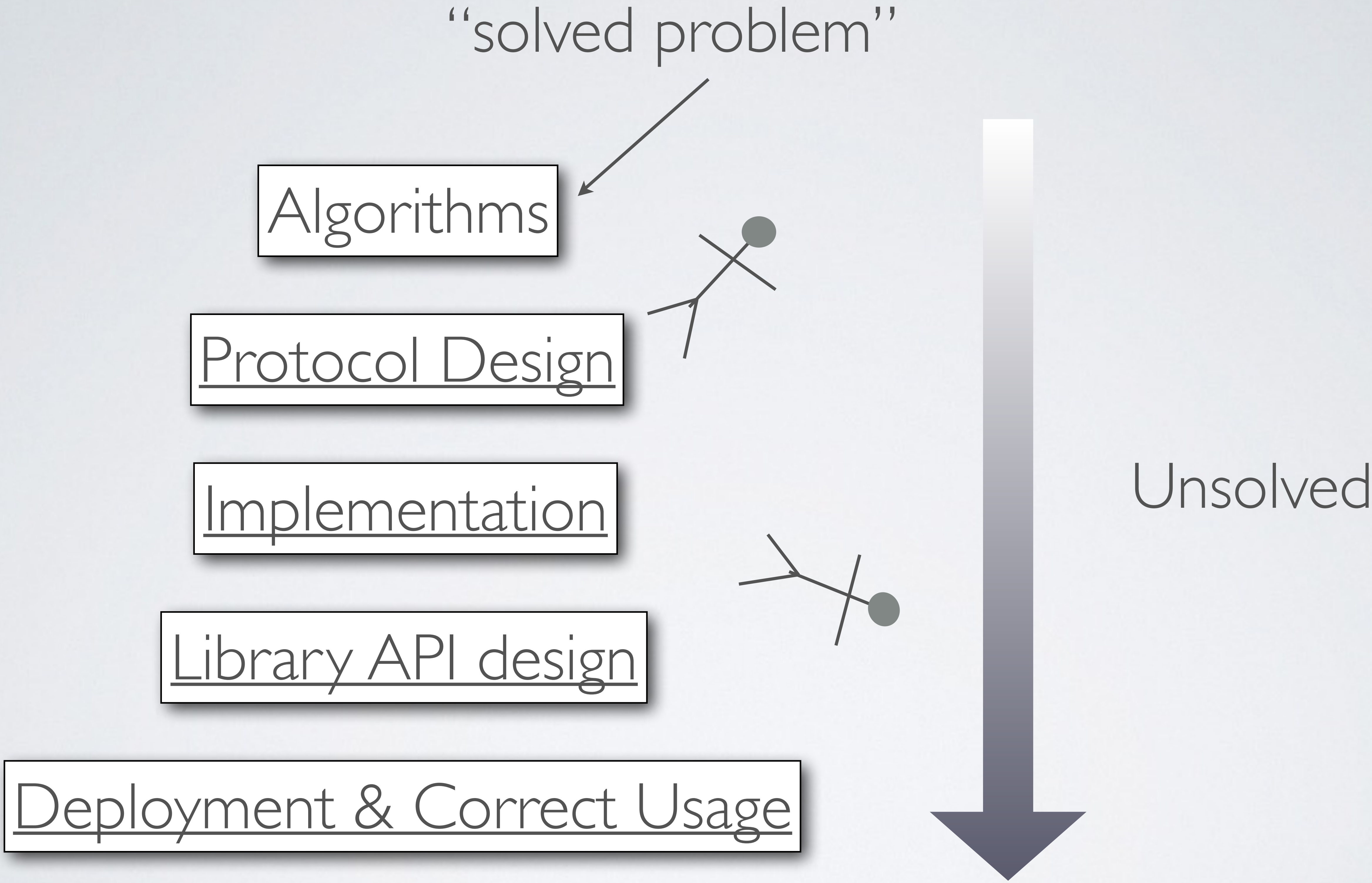
5: Cryptog Solved Problem

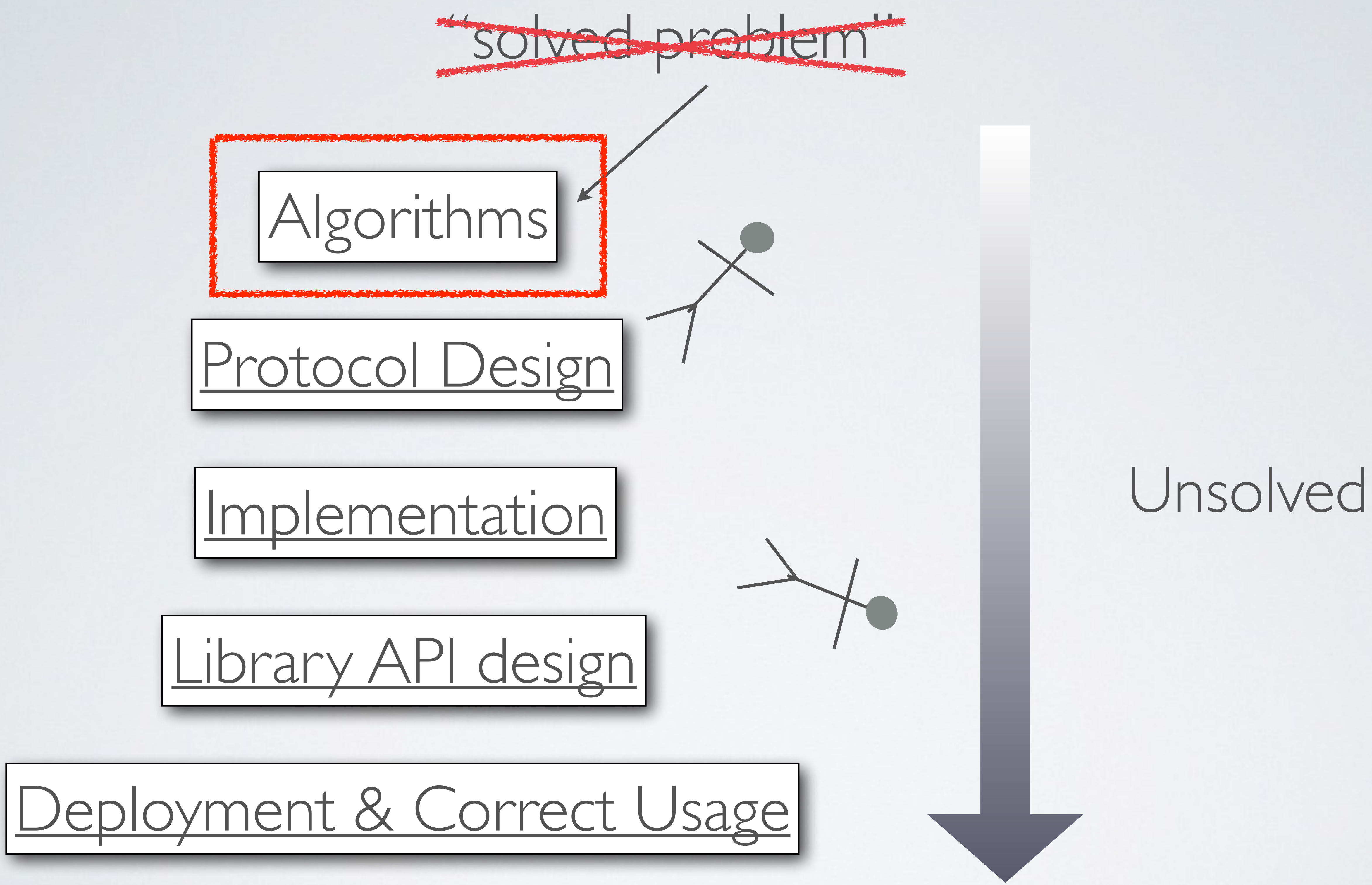
Cryptography: Th t link in the chain*

but not to others. Unfortunately, people concentrate on the cryptography of a system – which is the equivalent of str the strongest link in a chain.

These people are wrong


"solved problem"
Algorithms
Protocol Design
Implementation
Library API design
Deployment & Correct Usage
Unsolved


"solved problem"
Algorithms
Protocol Design
Implementation
Library API design
Deployment & Correct Usage
Unsolved


~~"solved problem"~~
Algorithms
Protocol Design
Implementation
Library API design
Deployment & Correct Usage
Unsolved

Why does this matter?

Download SuperFish Removal Tool

goto fail; // Apple SSL bug test site

This site will help you determine whether your computer is vulnerable to #goto

Tracking the FREAK Attack

QUALYS® SSL LABS

LOGJAM ATTACK (CVE-2015-4000)

TLS Vulnerability

Response to improving security

- For the past decade, NSA has lead an aggressive, multi-pronged effort to break widely used Internet encryption technologies
- Cryptanalytic capabilities are now coming on line
- Vast amounts of encrypted Internet data which have up till now been discarded are now exploitable
- Major new processing systems, SIGDEV efforts and tasking must be put in place to capitalize on this opportunity

PTD “We penetrate targets’ defences.”

GCHQ

1. **Highly Efficient GF(2^8) Inversion Circuit Based on Redundant GF Arithmetic and Its Application to AES Design**
   Rei Ueno (Tohoku University); Naofumi Homma (Tohoku University); Yukihiro Sugawara (Tohoku University); Yasuyuki Nogami (Okayama University); Takafumi Aoki (Tohoku University)
2. **Robust Profiling for DPA-Style Attacks**
   Carolyn Whitnall, Elisabeth Oswald (University of Bristol)
3. **SoC it to EM: electromagnetic side-channel attacks on a complex system-on-chip**
   Jake Longo (University of Bristol); Elke De Mulder (Cryptography Research Inc.); Dan Page (University of Bristol); Michael Tunstall (Cryptography Research Inc.)
4. **TriviA: A Fast and Secure Authenticated Encryption Scheme**
   Avik Chakraborti (Indian Statistical Institute Kolkata); Anupam Chattopadhyay (School of Computer Engineering, NTU Singapore); Muhammad Hassan (RWTH Aachen University); Mridul Nandi (Indian Statistical Institute Kolkata)
5. **Stealing Keys from PCs using a Radio: Cheap Electromagnetic Attacks on Windowed Exponentiation**
   Daniel Genkin (Technion and Tel-Aviv University); Lev Pachmanov (Tel-Aviv University); Itamar Pipman (Tel-Aviv University); Eran Tromer (Tel-Aviv University)
6. **Efficient Ring-LWE Encryption on 8-bit AVR Processors**
   Zhe Liu (University of Luxembourg); Hwajeong Seo (Pusan National University); Sujoy Sinha Roy (K.U. Leuven); Johann Großschädl (University of Luxembourg); Howon Kim (Pusan National University); Ingrid Verbauwhede (K.U. Leuven)

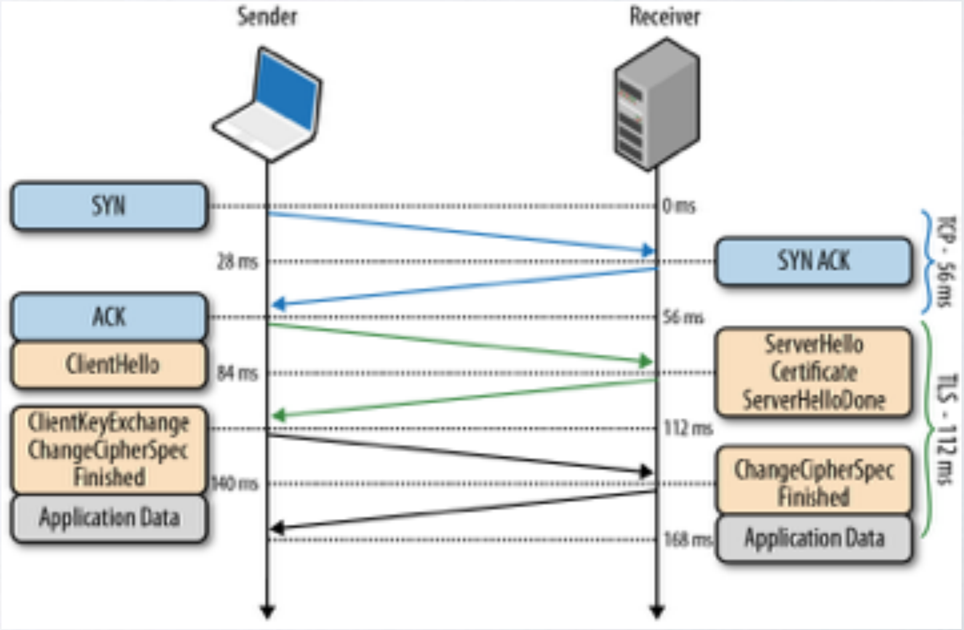
This talk

- We know how to build strong cryptosystems
  - Our research focuses on building <u>stronger</u> crypto systems!
- And yet we continue to deploy weak ones
  - Worse, we're largely stuck with weak ones
  - What's going on here?
  - **Main case studies: SSL/TLS, IPSEC**

Case study 1: SSL/TLS

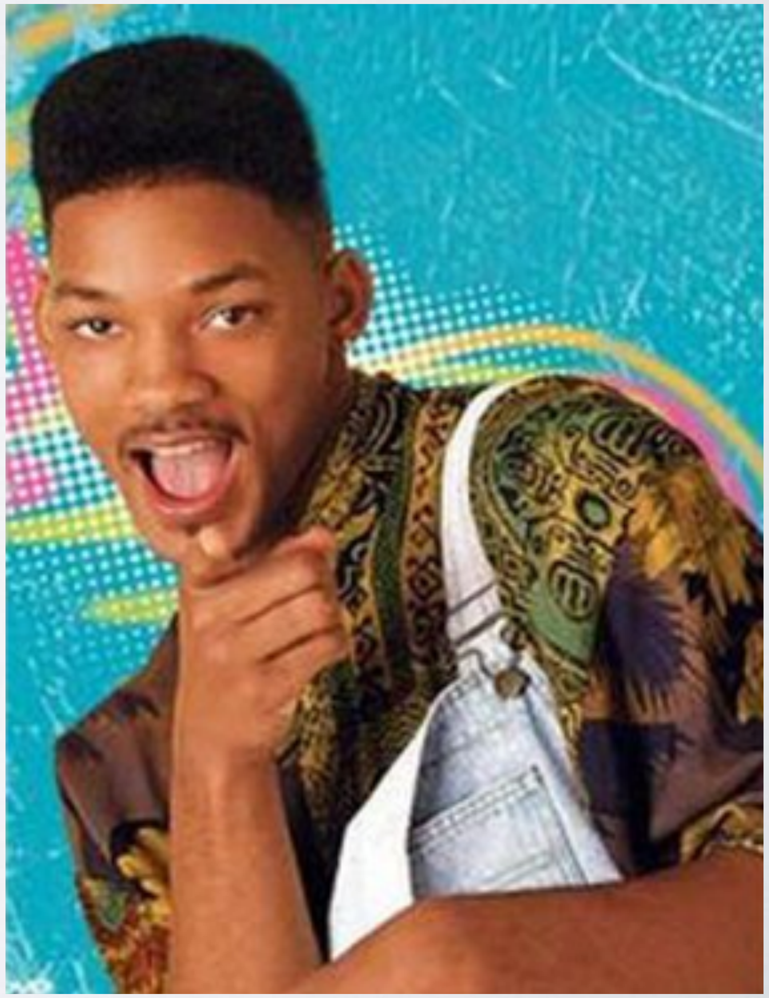
SSL/TLS

- **Most important security protocol on the Internet**
  - Allows secure connections between clients & servers
  - Current version: TLS 1.2
    - (But browsers still support SSL 3, TLS 1.0/1.1) plus 1.3 coming soon!
  - Not just web browsing!



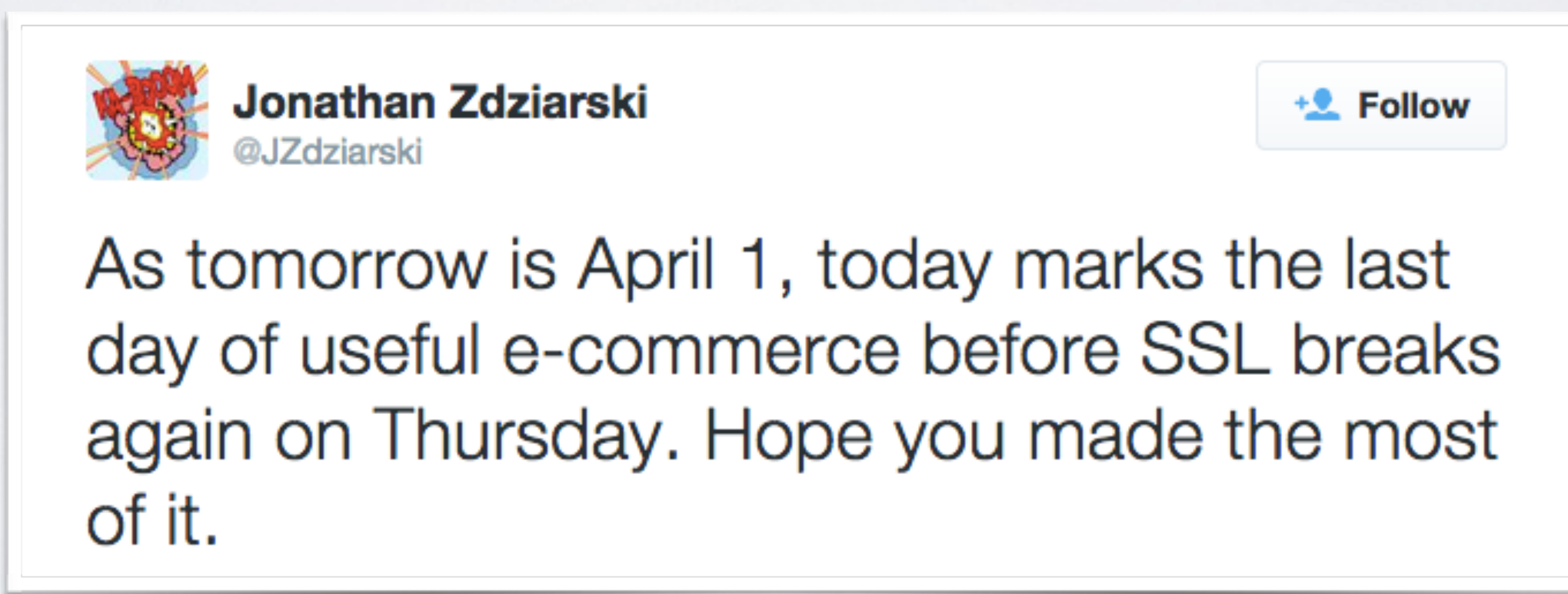
A brief history

- **SSLv1 born at Netscape. Never released. (~1994)**
- **SSLv2 released one year later**
- **SSLv3 (1996)**
- **TLS 1.0 (1998)**
  - Still widely deployed
- **TLS 1.1 (2006)**
- **TLS 1.2 (2008)**



How secure is TLS?

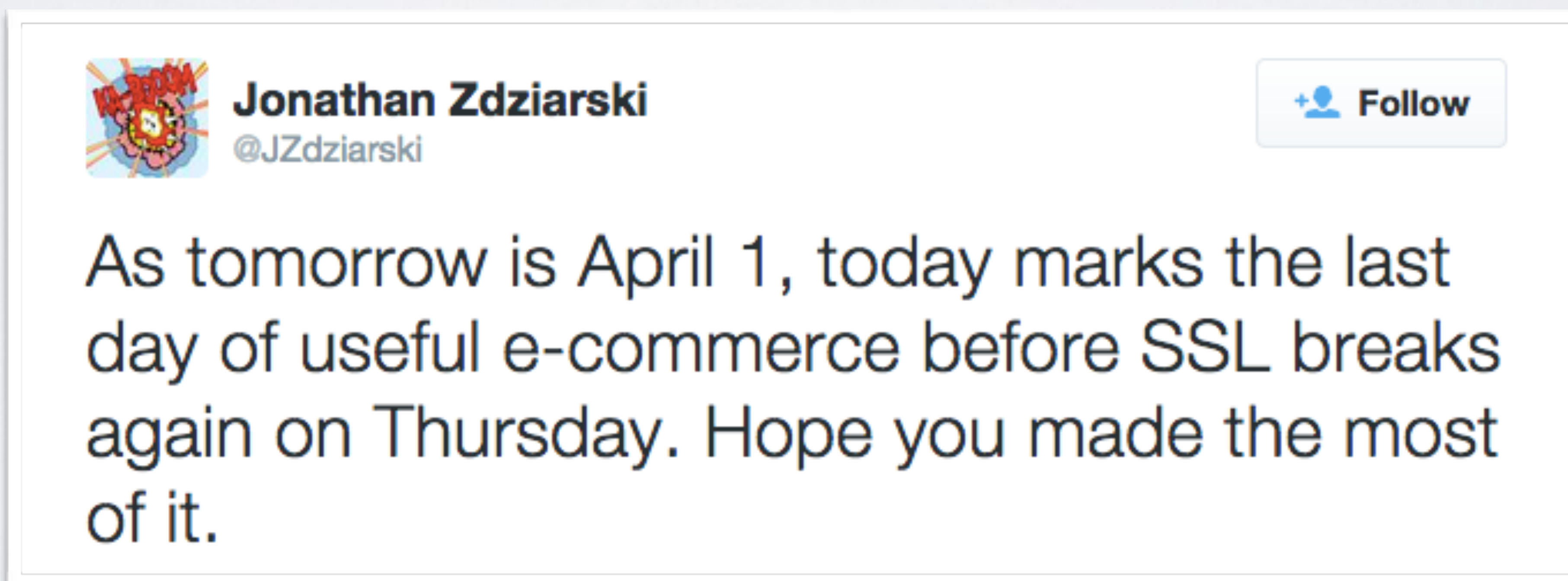
- **Many active attacks and implementation vulnerabilities**
  - Heartbleed, Lucky13, FREAK, CRIME, BEAST, RC4



How secure is TLS?

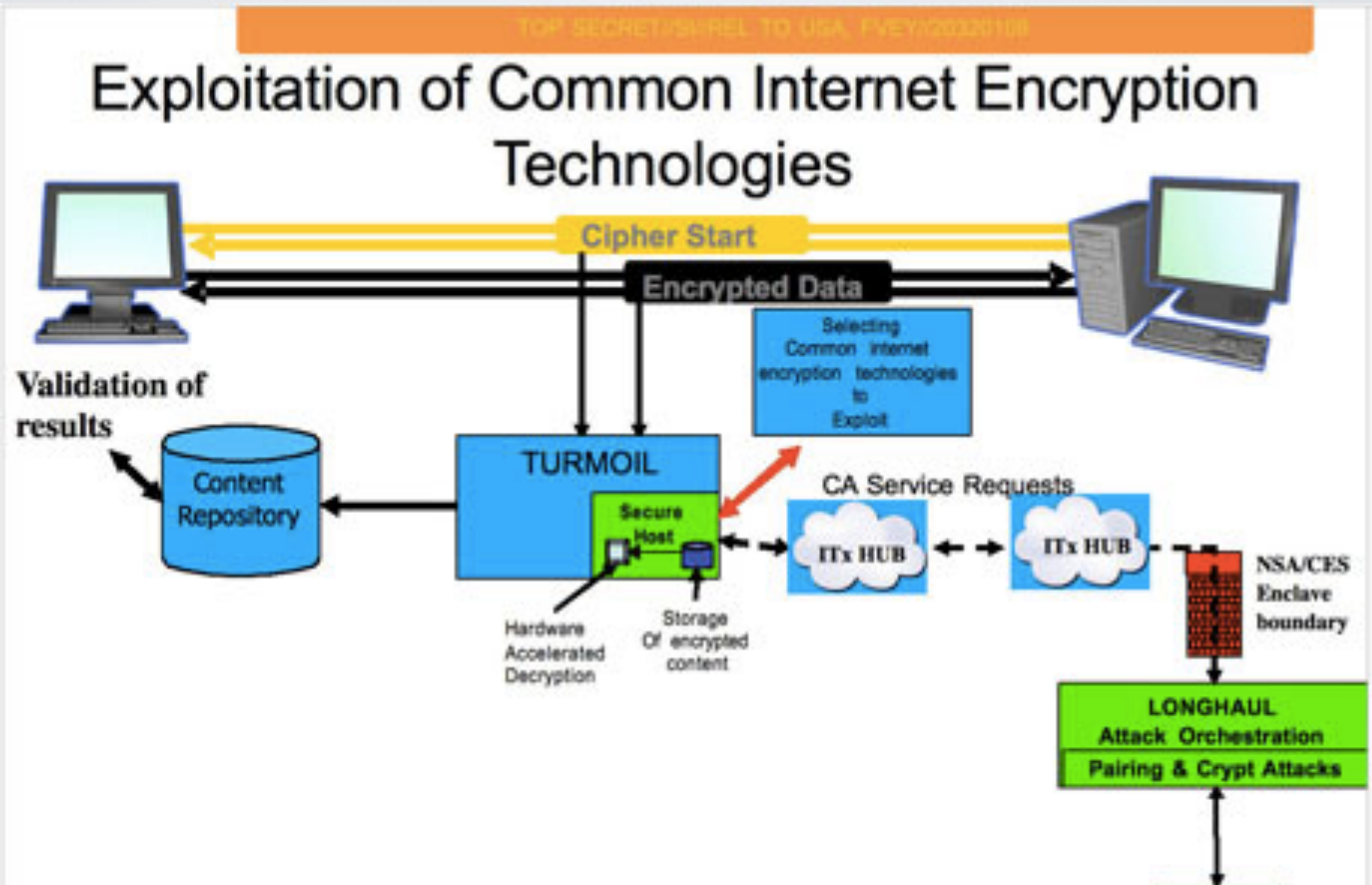
- **Many active attacks and implementation vulnerabilities**
  - Heartbleed, Lucky13, FREAK, CRIME, BEAST, RC4

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

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

BEASTT

- 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)

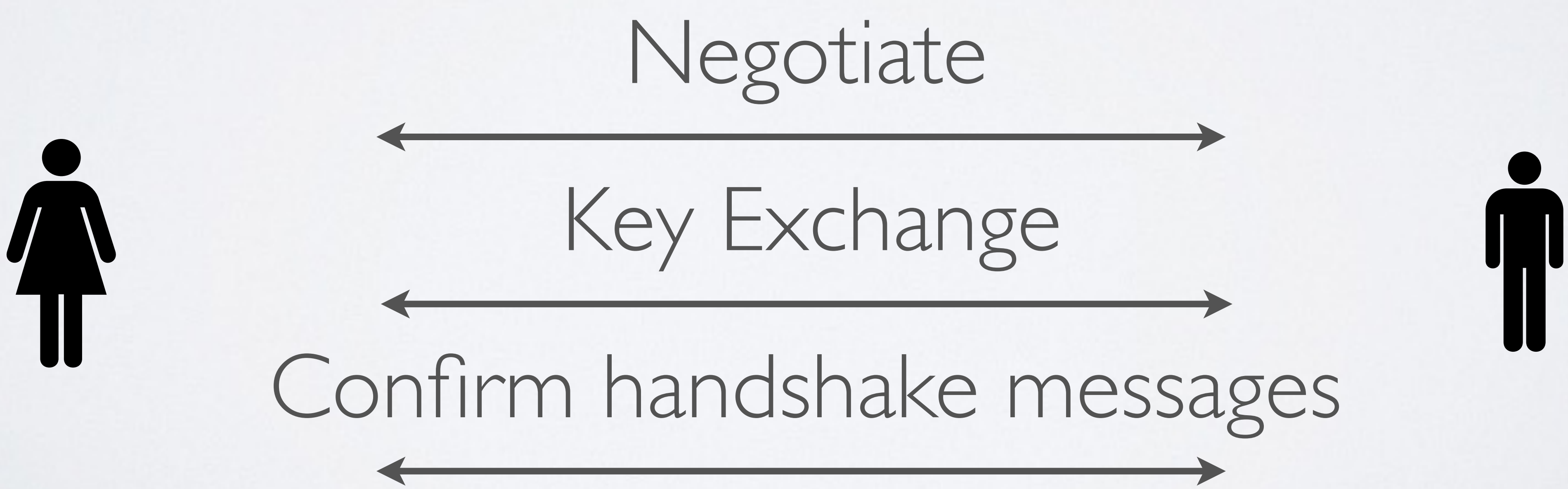
Compression (CRIME)

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

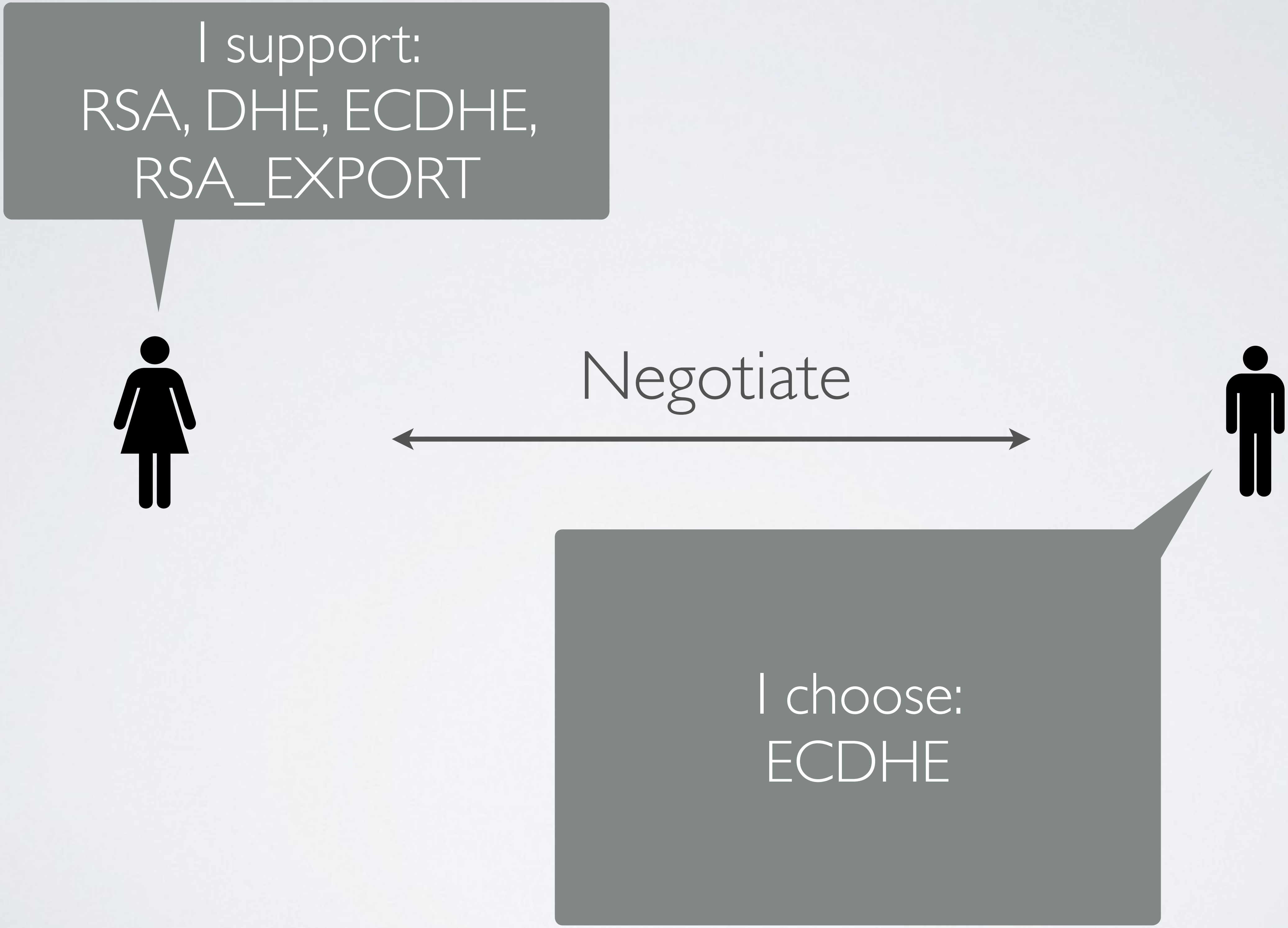
Protocol Design

Example: Negotiation

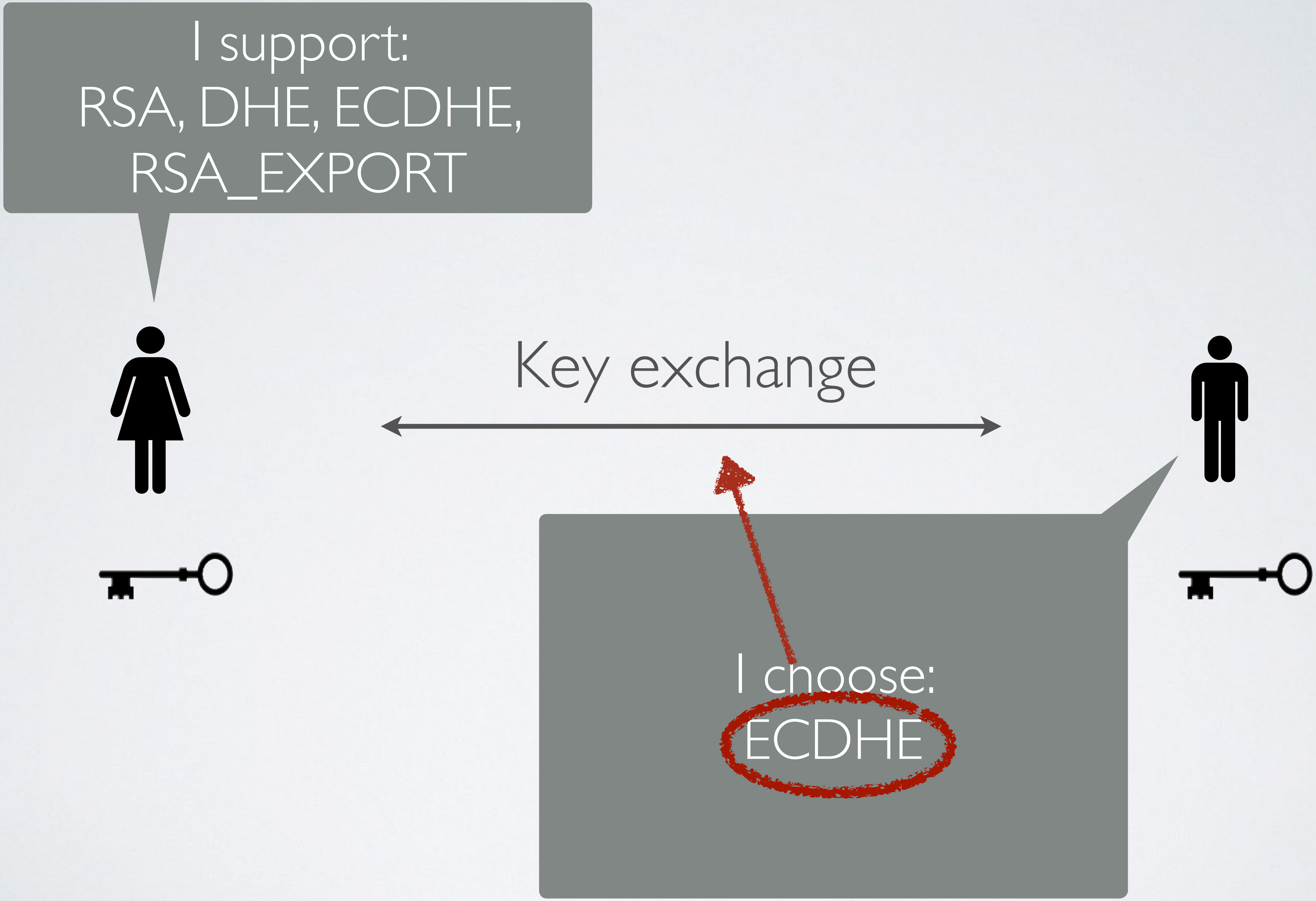
Each TLS handshake begins with a cipher suite negotiation that determines which key agreement protocol (etc.) will be used.



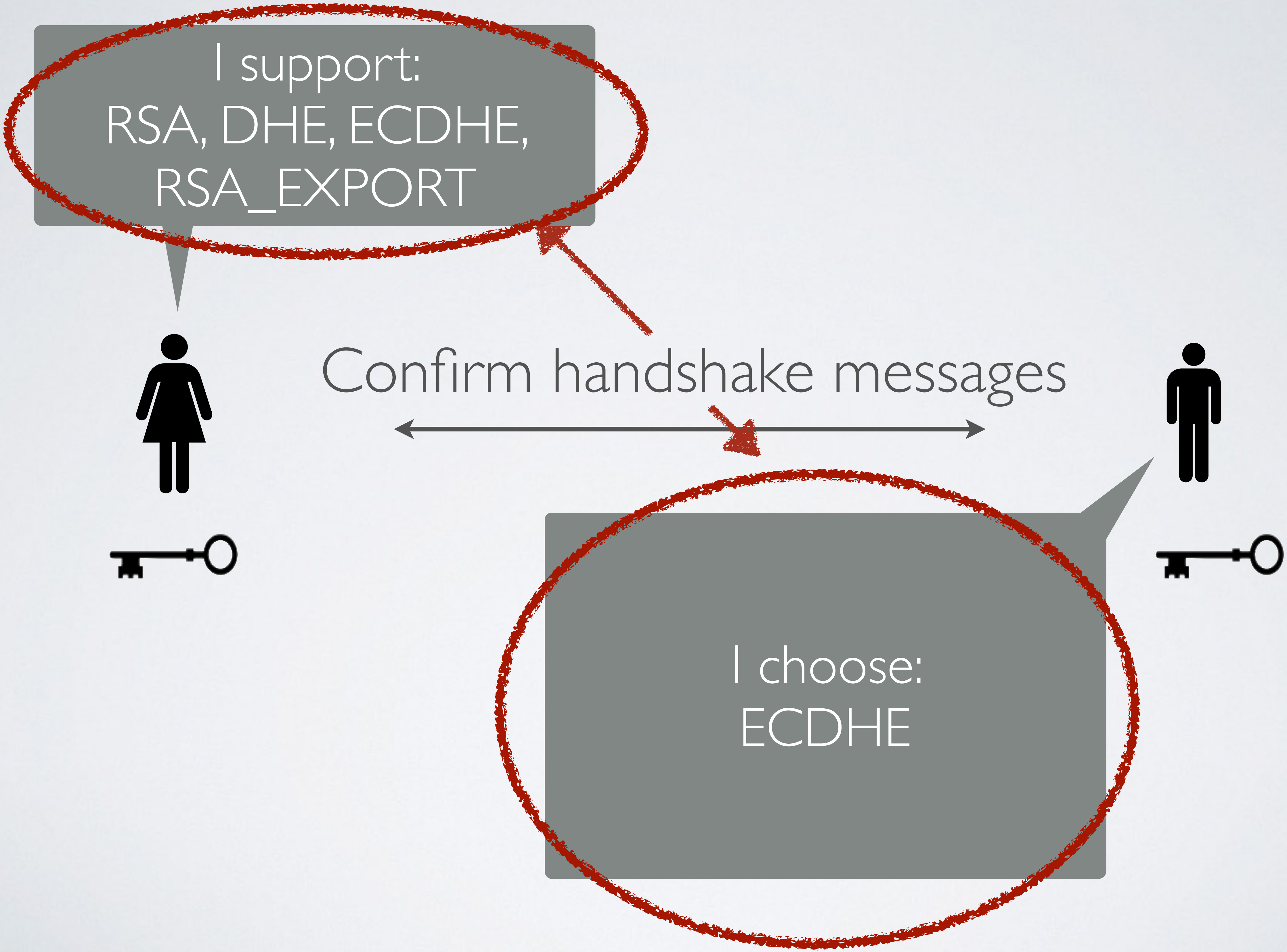
Ciphersuite Negotiation



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:

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.

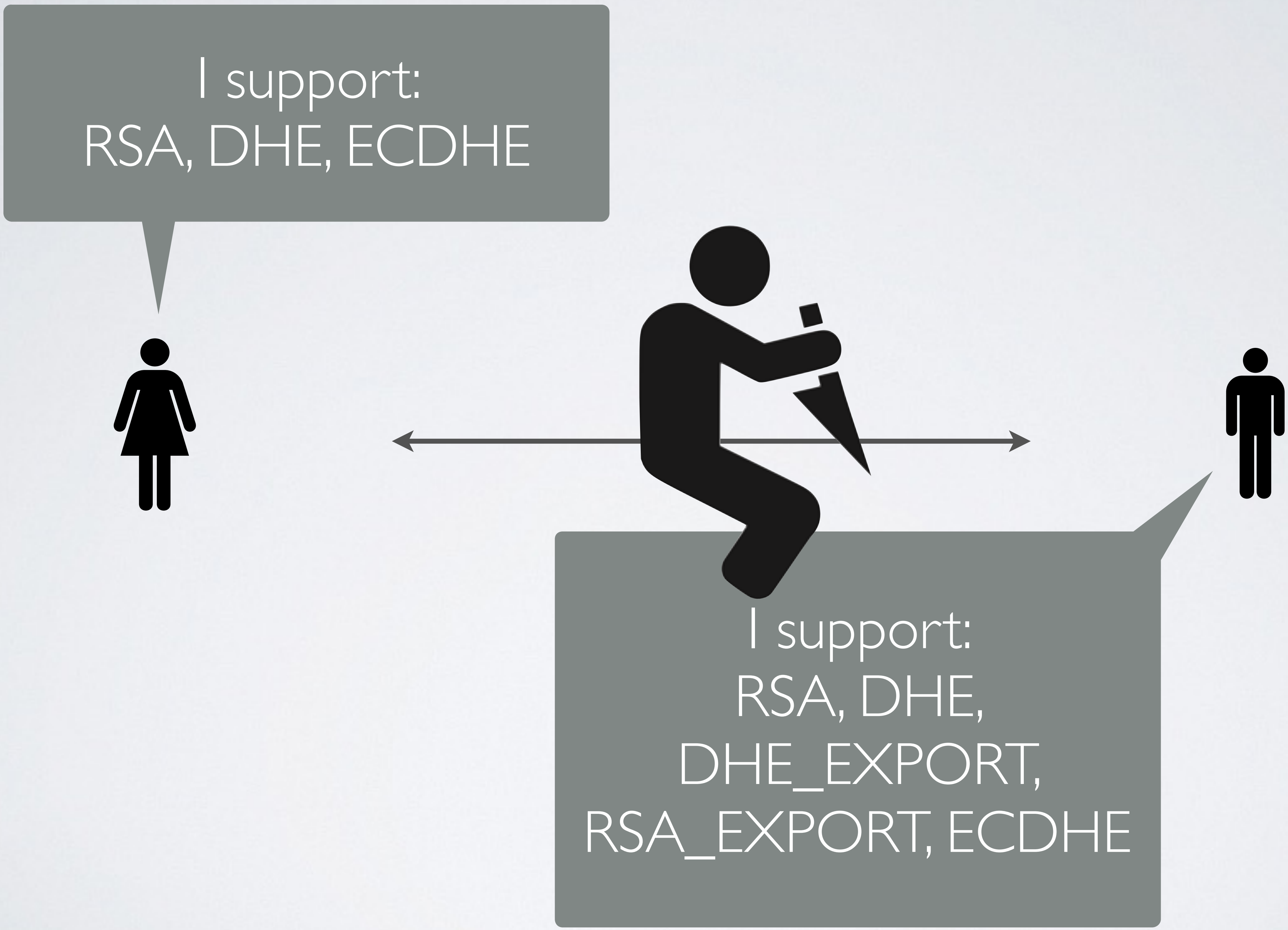
Theorem

- **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.

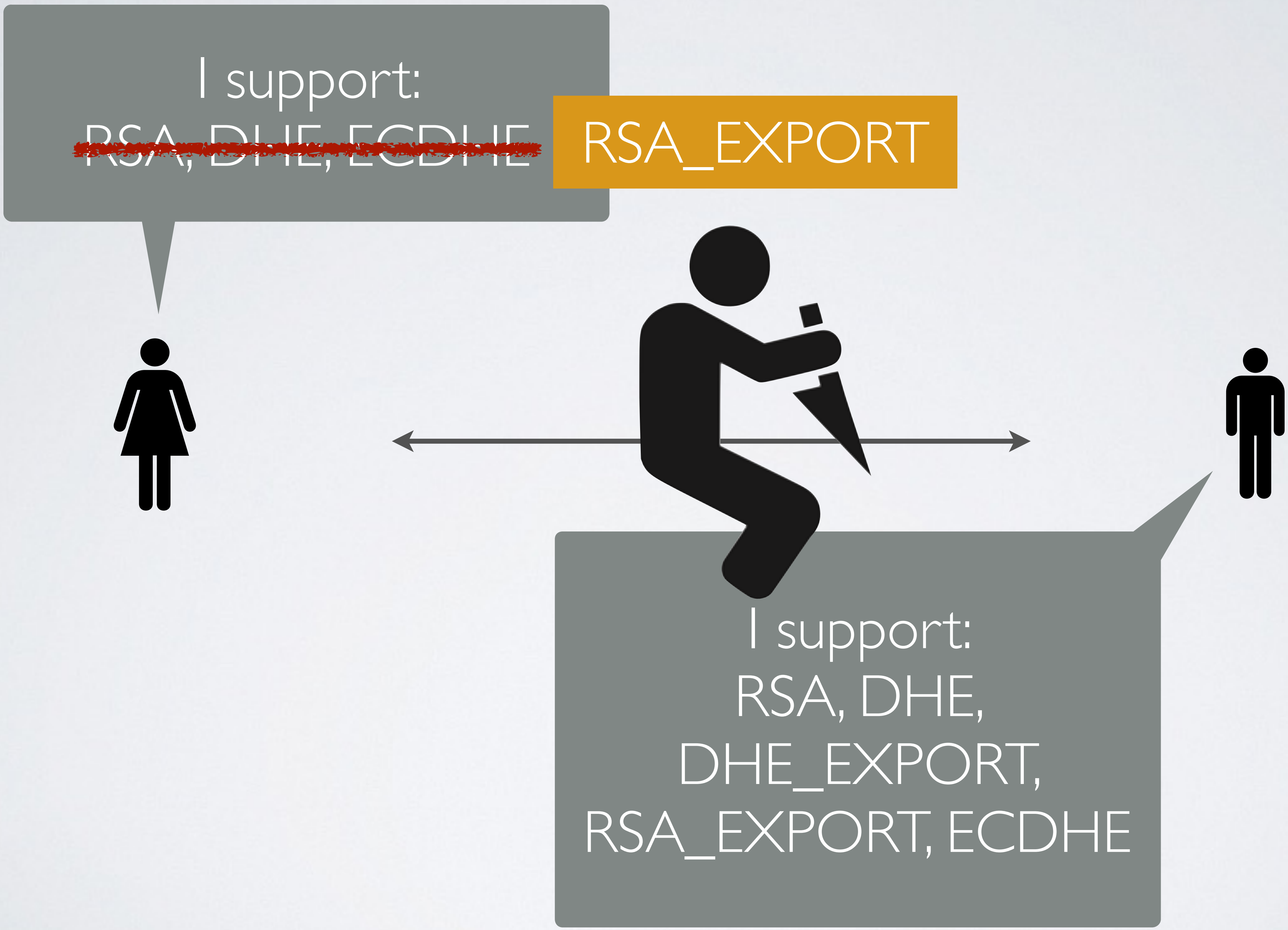
TLS design/deployment assumes this would be $A \cap B$!

Example 2: Negotiation



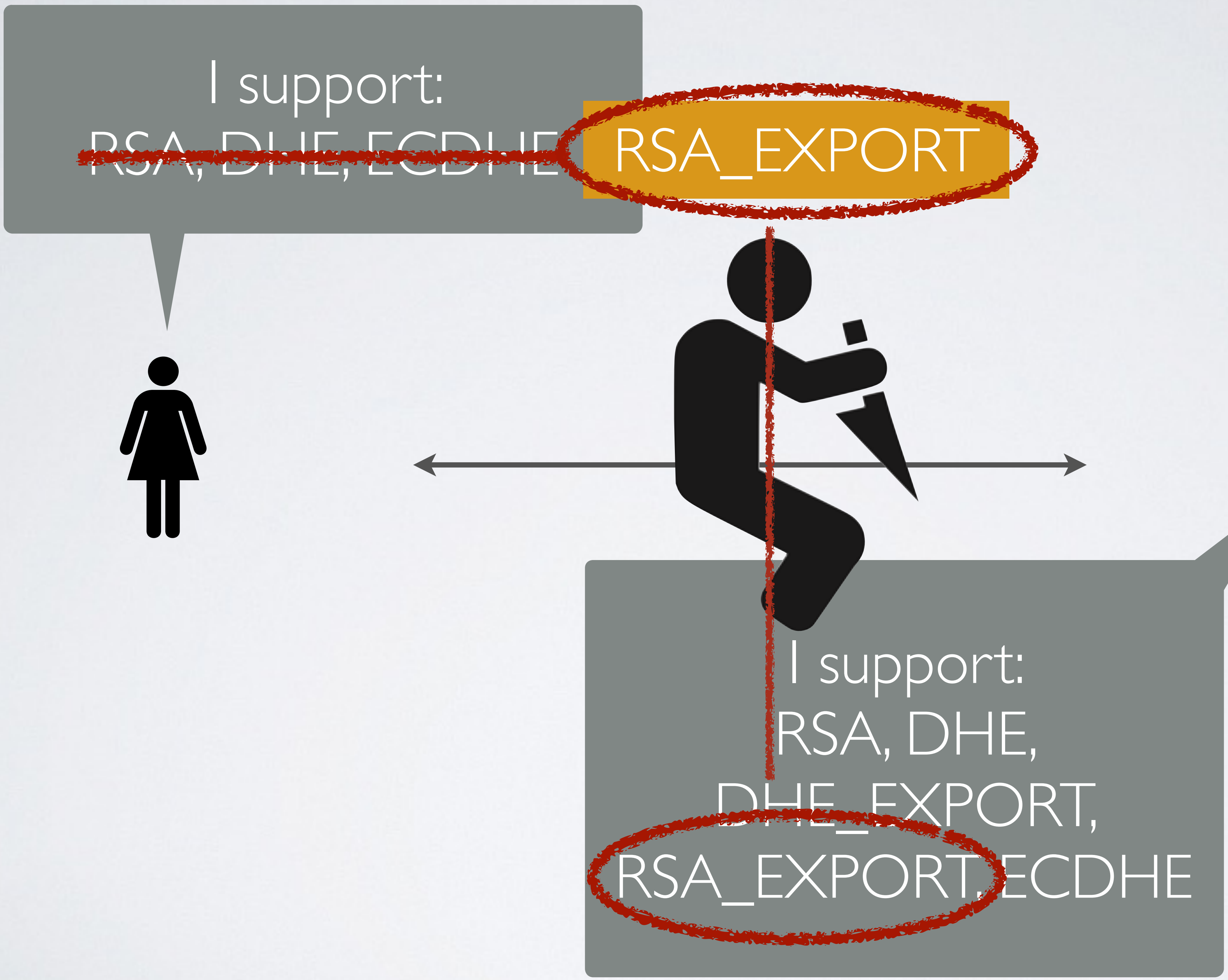
Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

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

I support:
~~RSA, DHE, ECDHE~~ RSA_EXPORT

FREAK [Bhargavan *et al.*]**:**
Due to a bug in SecureTransport,
OpenSSL, SChannel,
client accepts export-grade RSA key

RSA, DHE,
DHE_EXPORT,
RSA_EXPORT, ECDHE

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 <u>recognize</u> an RSA export key exchange message, barf

(patches rolled out March 2015)

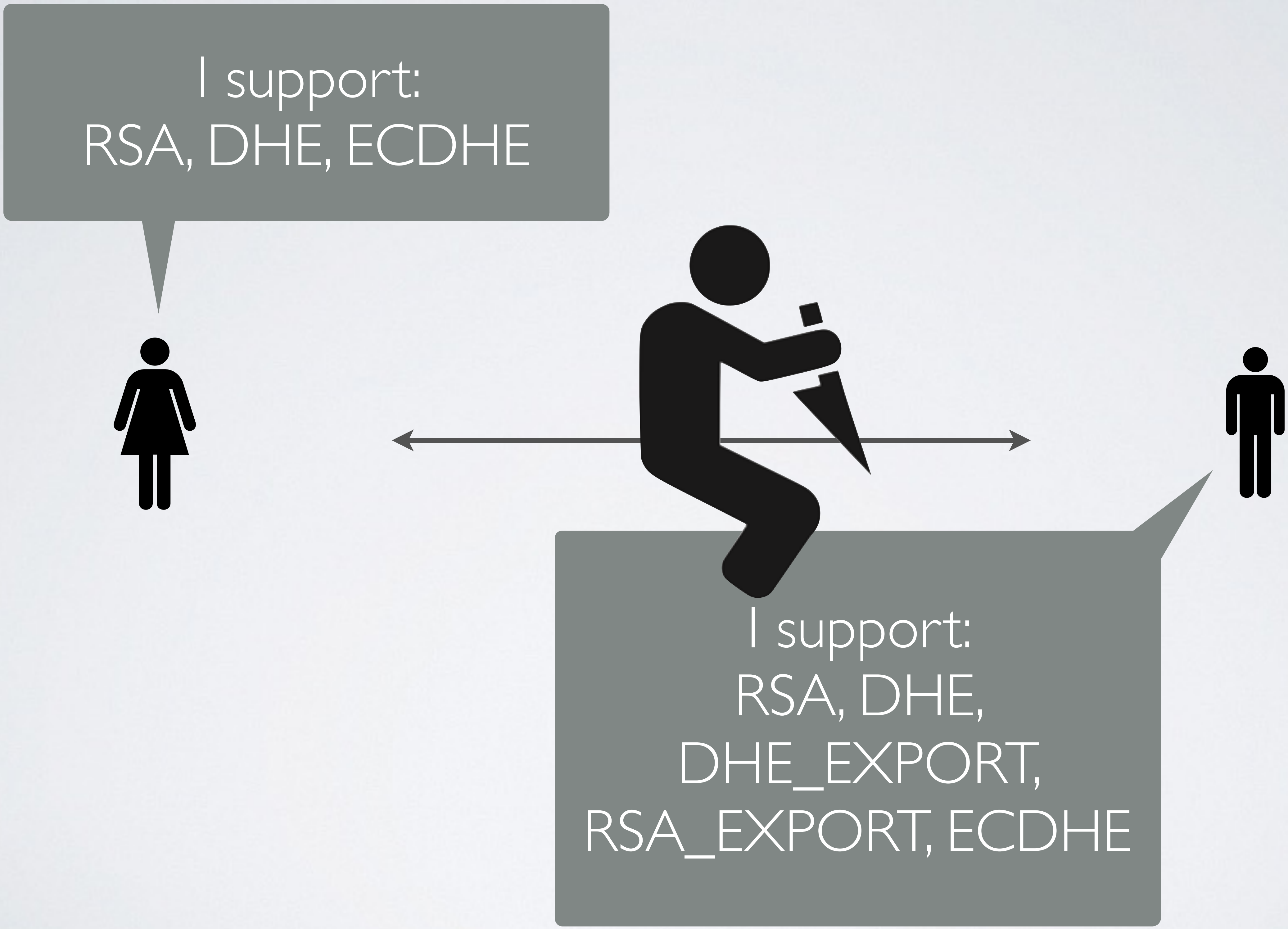
RSA, DHE

Apple issues security patches to protect devices from the FREAK bug

by Mariella Moon | @mariella_moon | 22 days ago

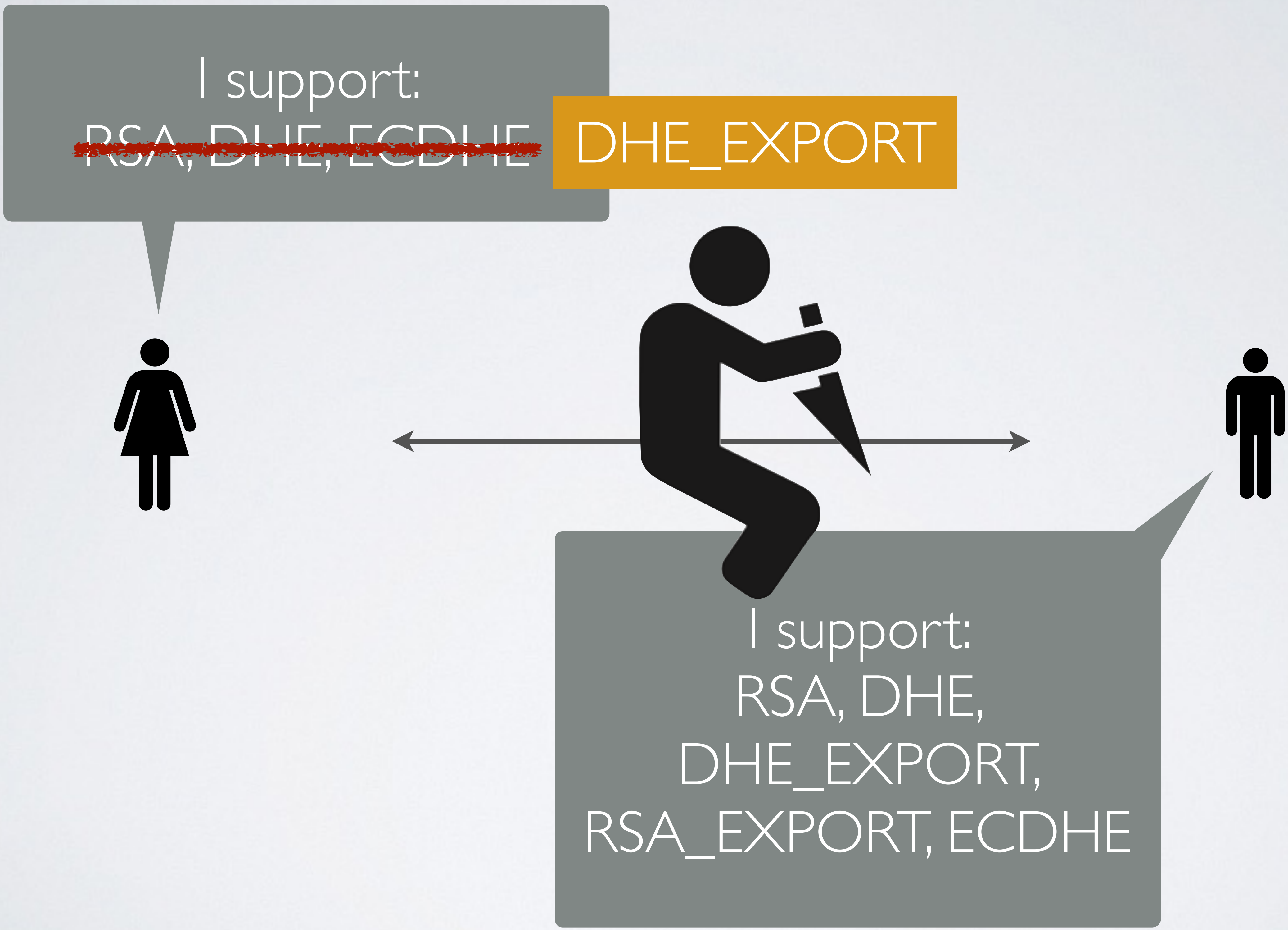
Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

Example 3: Negotiation



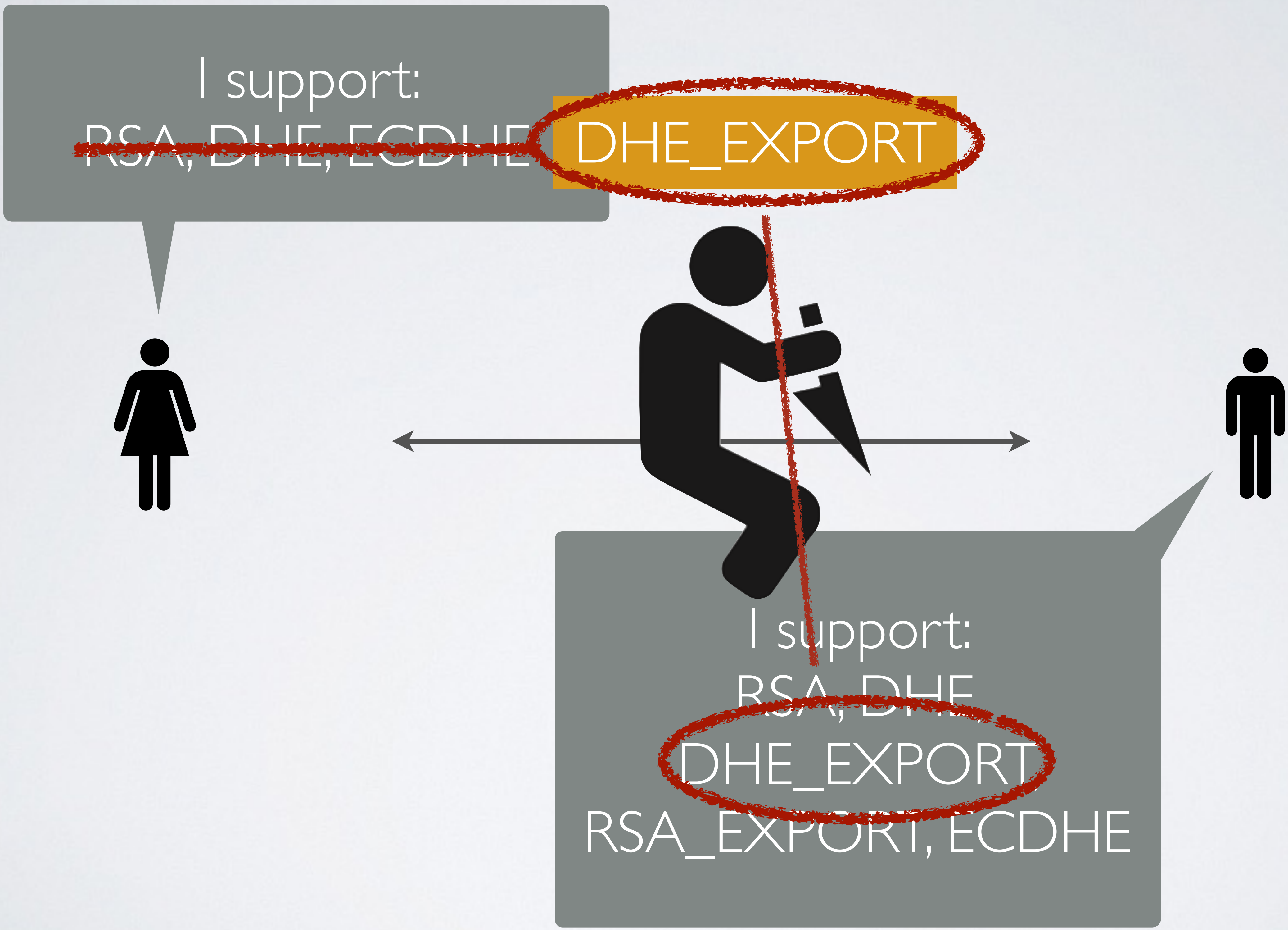
Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash, Cryptographic Agility and its Relation to Circular Encryption, in EUROCRYPT 2010

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

I support:
~~RSA, DHE, ECDHE~~ DHE_EXPORT

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

RSA, DHE
DHE_EXPORT
RSA_EXPORT, ECDHE

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 <u>only responded to attacks.</u>

Exploiting LogJam

(Joint work: Adrian, Bhargavan, Durumeric, Gaudry, Green, Halderman, Heninger, Springall, Thomé, Valenta, VanderSloot, Wustrow, Zanella-Beguelin, Zimmermann) *to appear* 'CCS 2015

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

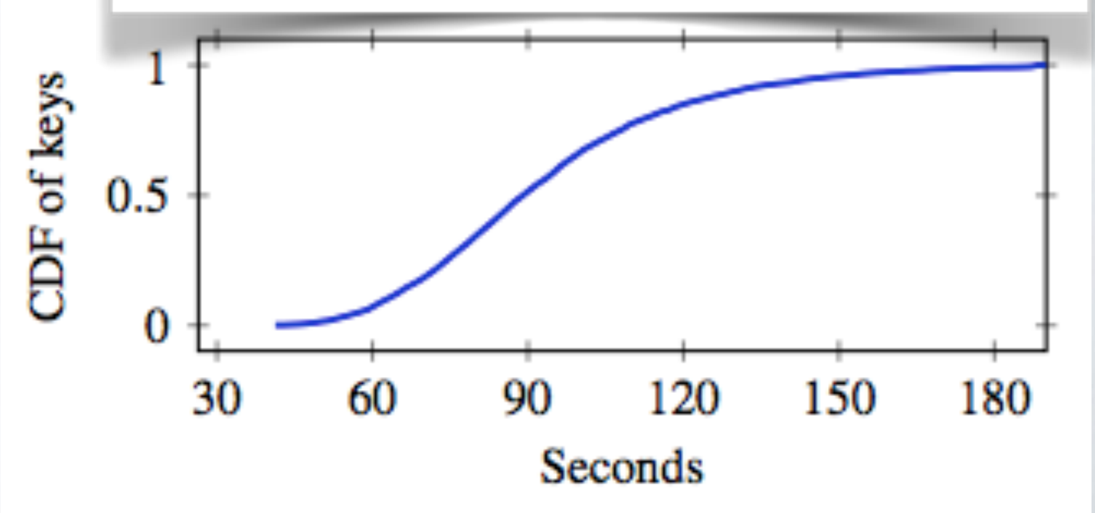
- To exploit the downgrade attack, requires solving a 512-bit DL in real time
    - **92% of DHE_EXPORT servers use one of two hard-coded primes (*p*) *(Mod_SSL, Apache)***

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

Server has a weak, ephemeral Diffie-Hellman public key

ERR_SSL_WEAK_SERVER_EPHEMERAL_DH_KEY

Hide details

This error can occur when connecting to a secure (HTTPS) server. It means that the server is trying to set up a secure connection but, due to a disastrous misconfiguration, the connection wouldn't be secure at all!

In this case, the server needs to be fixed. Google Chrome won't use insecure connections in order to protect your privacy.

Learn more about this problem.

Long(er) term solutions:

Eliminate 1024-bit DHE (but Java).

Stop using common DHE primes.

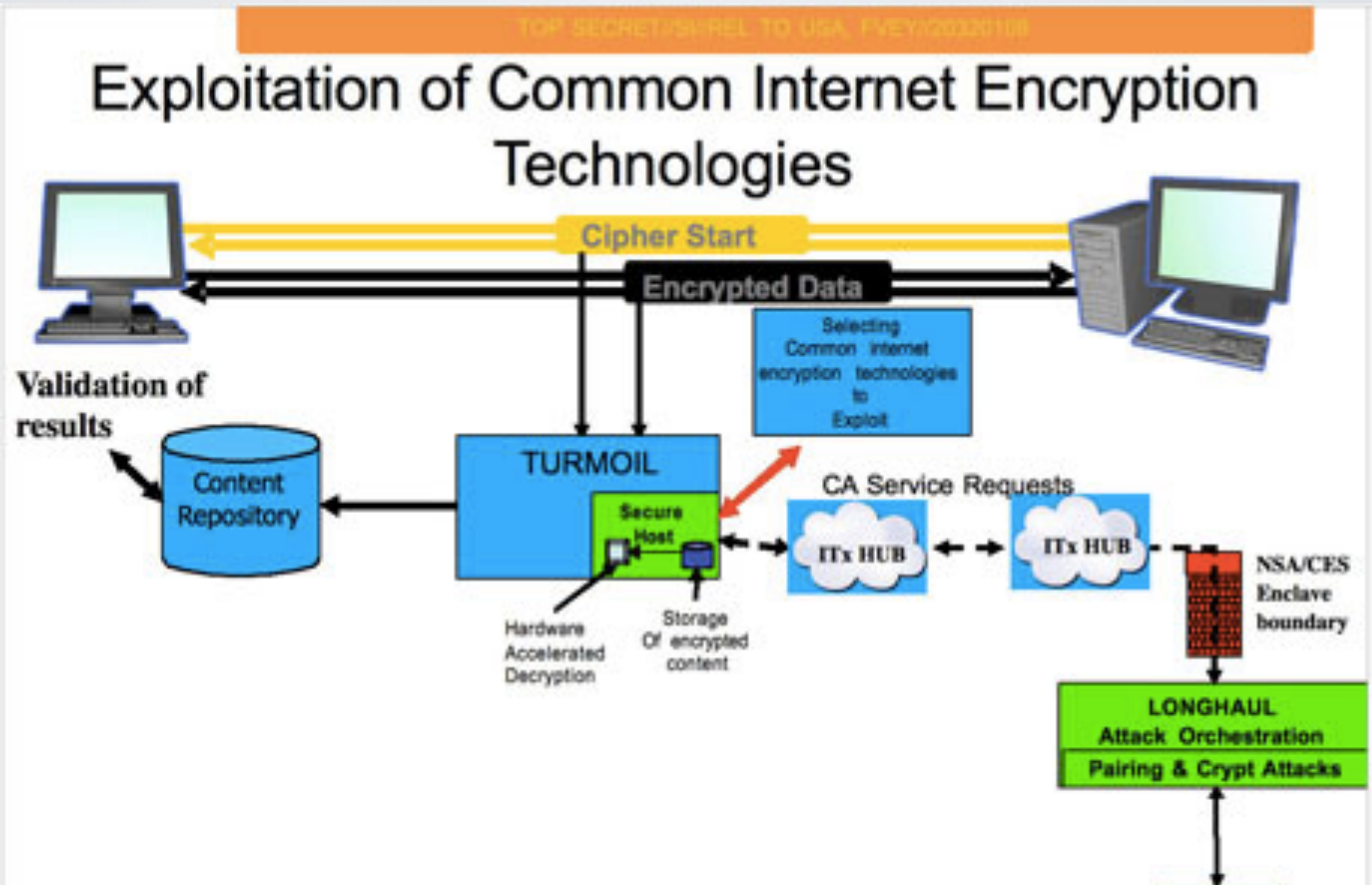
Use EU-CMA signatures to validate the protocol transcript. Then you can achieve the $A \cap B$ security the TLS designers originally set out to achieve.

(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

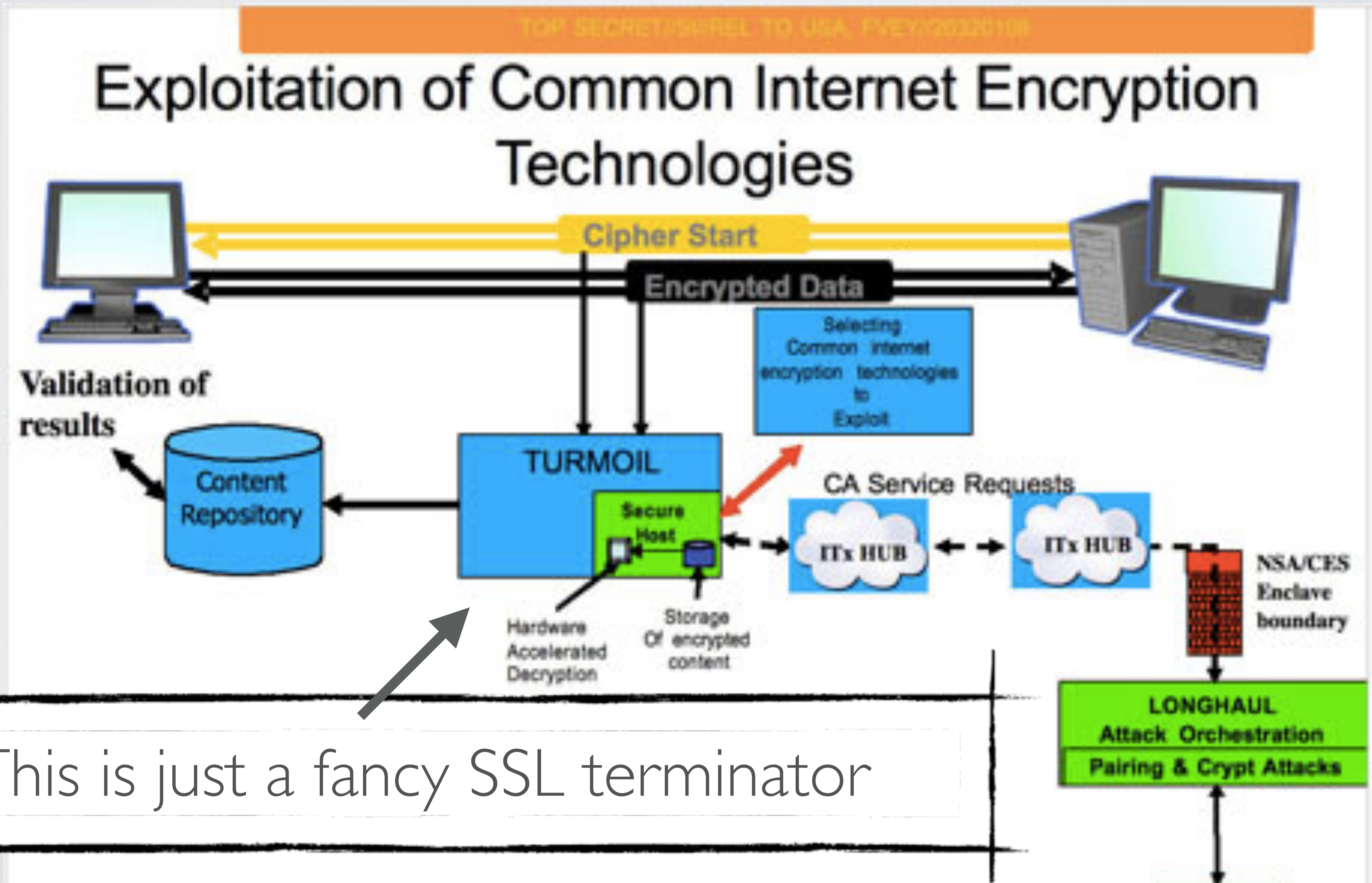
This picture again

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



This picture again

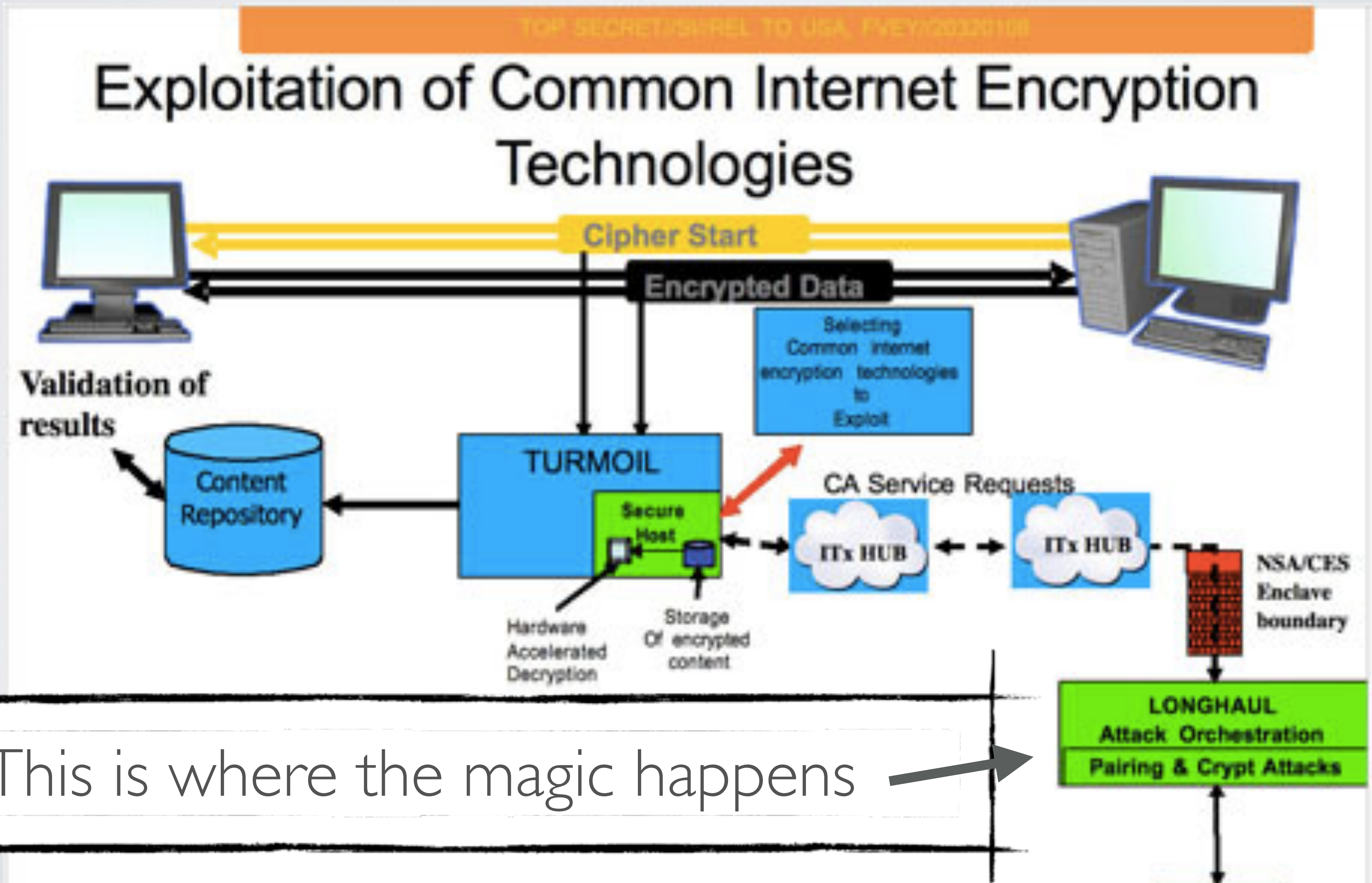
- **What's going on here?**



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

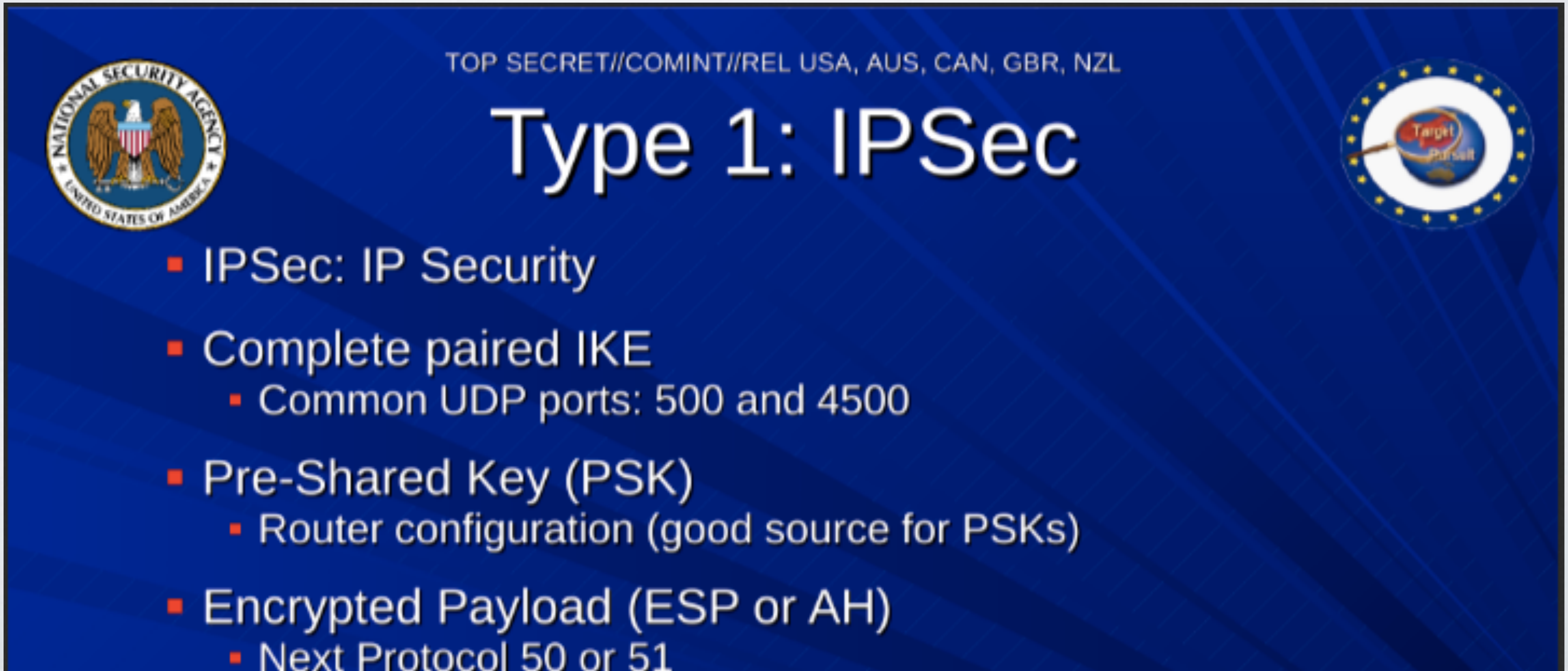
DECRYPTION!

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

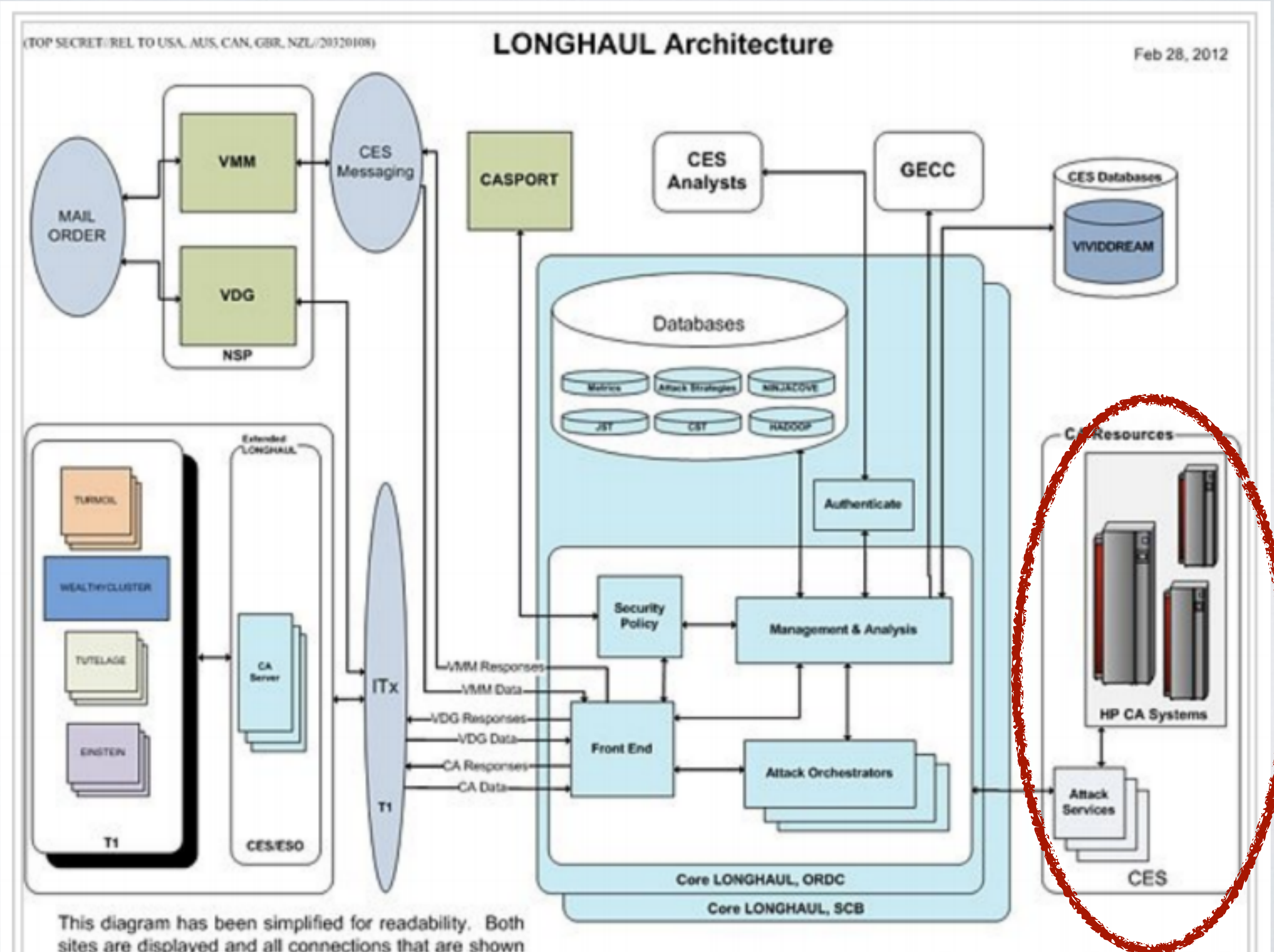
Happy Dance!!

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 <u>with academic levels of computing</u>**
  - 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

Why aren't we fixing this?

Why aren’t we fixing this?

Adrienne Porter Felt
@__apf__

Follow

Large chunks of the internet have rotted with age. They aren't updated or updateable. Wtf do we do?

RETWEETS	FAVORITES
21	33

6:54 AM - 5 Sep 2015

Conclusion

- Cryptography is challenging! (duh)
  - 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