

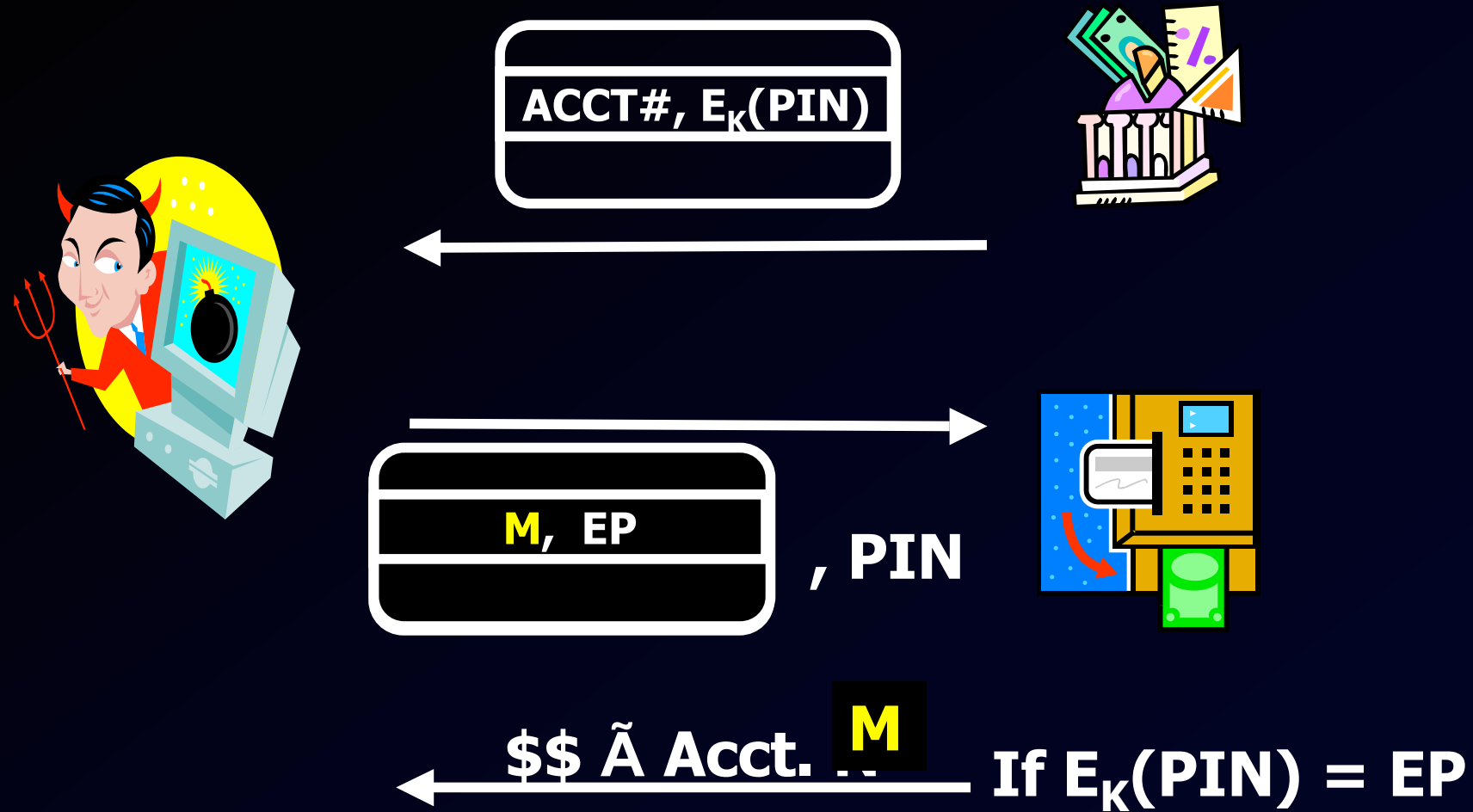
UMSSIA

DAY VI: ARE WE THERE YET?

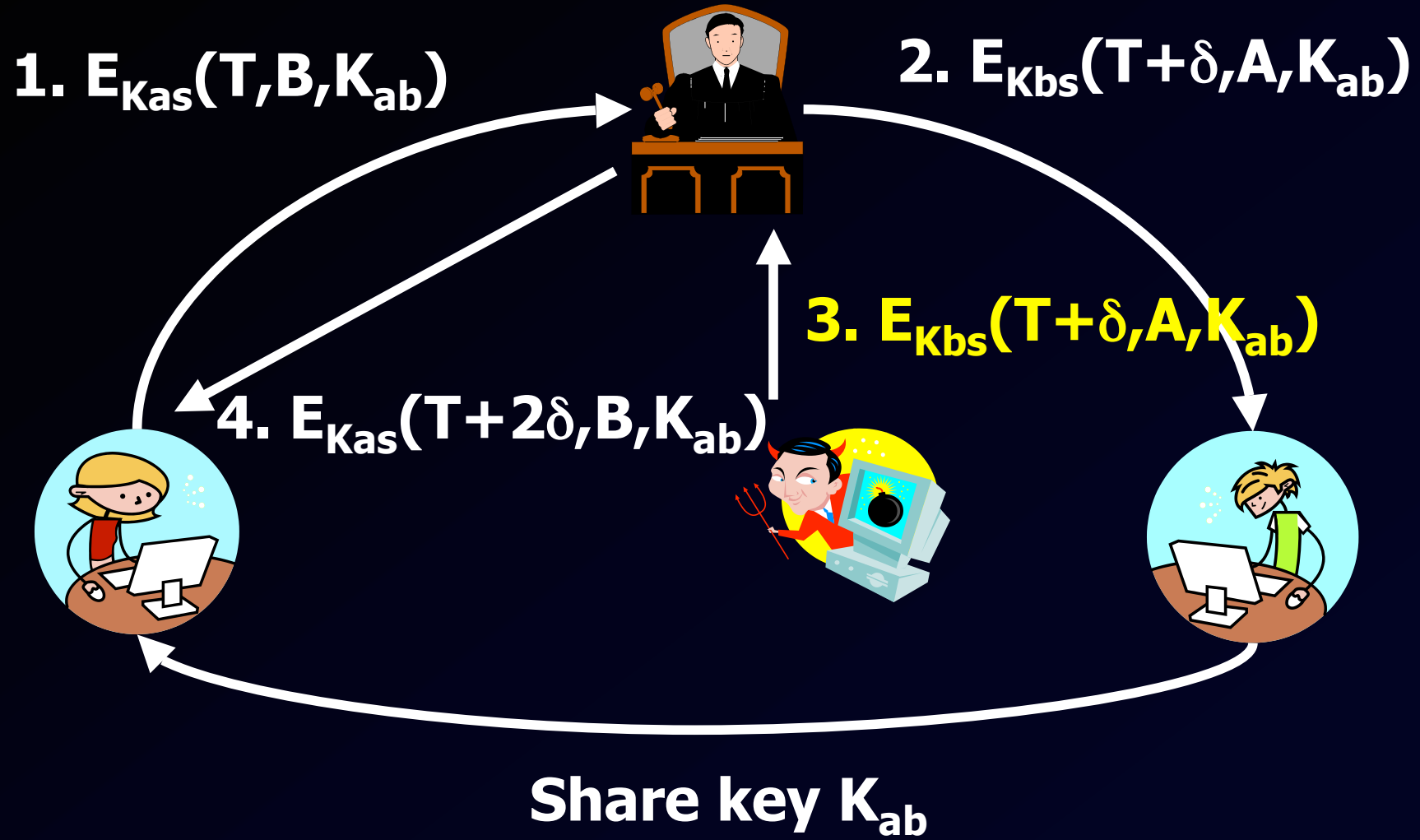
CRYPTO PROTOCOLS

- **Good crypto algorithms are hard to design but easy to find on the web.**
- **Building robust security protocols, even from secure algorithms, is also hard.**
- **Subtle bugs can be found 10+ years after public scrutiny. Anderson and Needham compare crypto protocol design to “Programming Satan’s Computer.”**

WARMUP



WIDE-MOUTHED FROG



WOO-LAM



1. Alice

2. N_B , a **Nonce**.

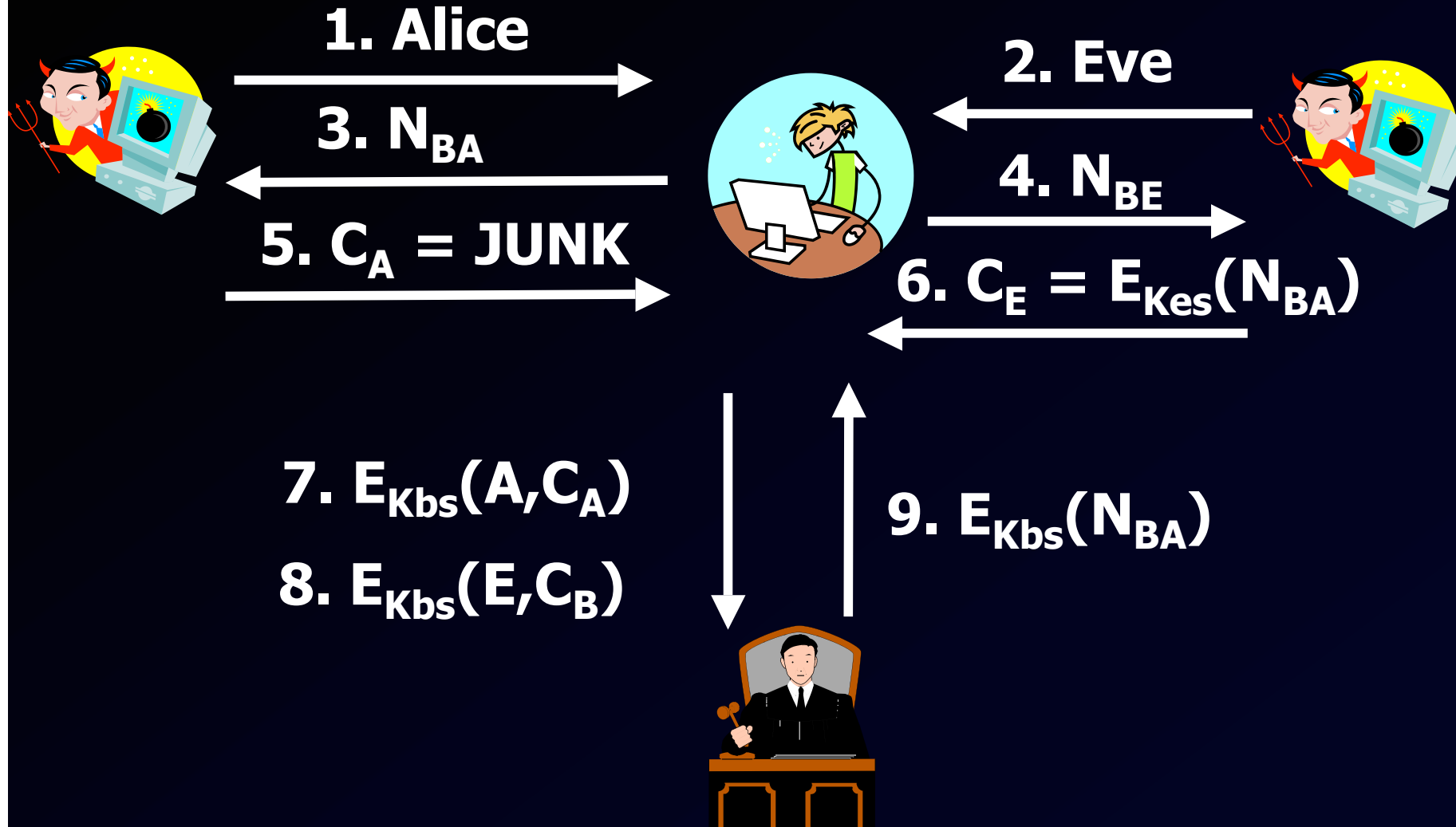
3. $C_A = E_{K_{as}}(N_B)$

4. $E_{K_{bs}}(A, C_A)$

5. $E_{K_{bs}}(N_B)$



ON THE WOO-LAM



WOO-LAM FIX?



1. Alice

2. N_B , a **Nonce**.

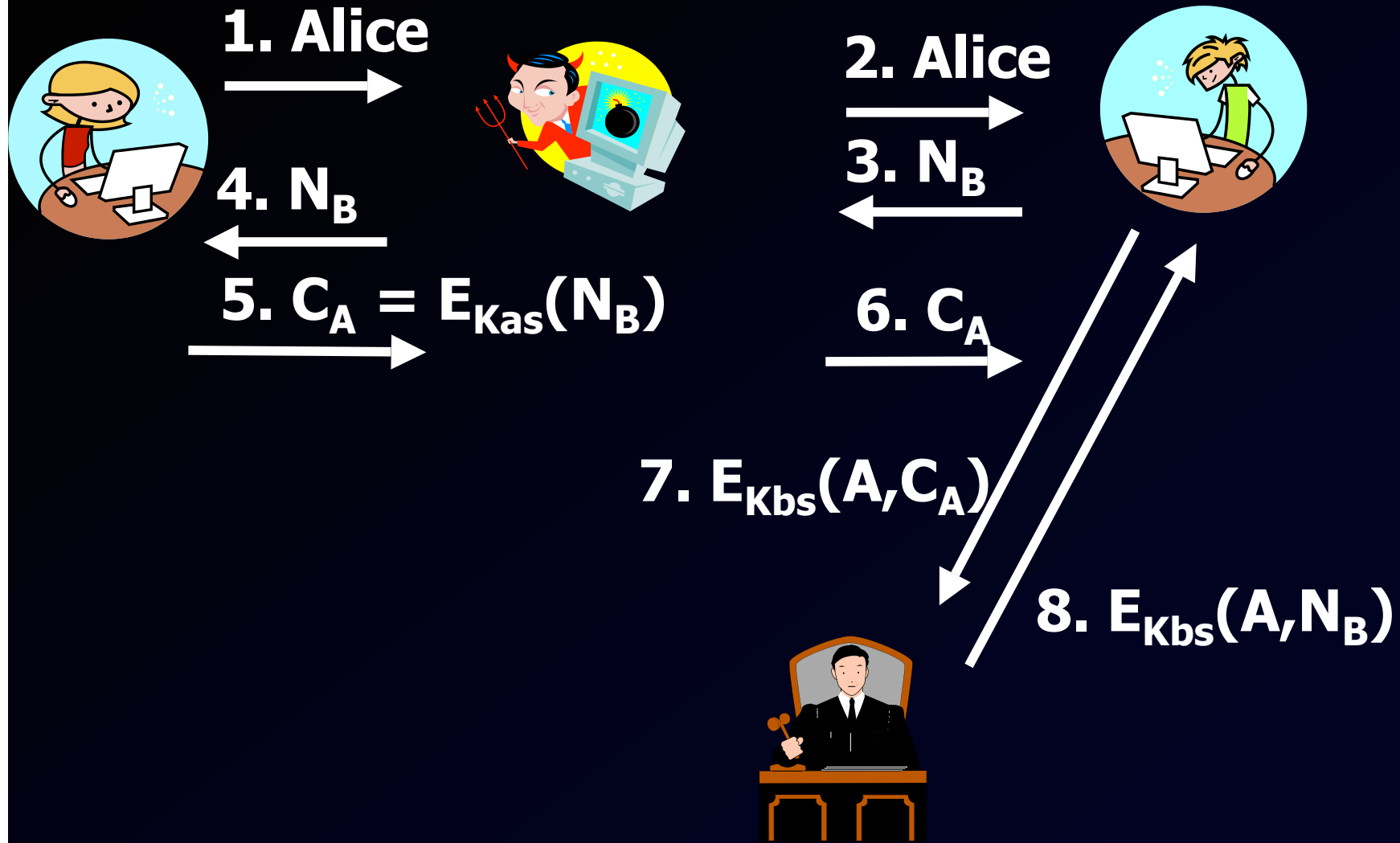
3. $C_A = E_{K_{as}}(N_B)$

4. $E_{K_{bs}}(A, C_A)$

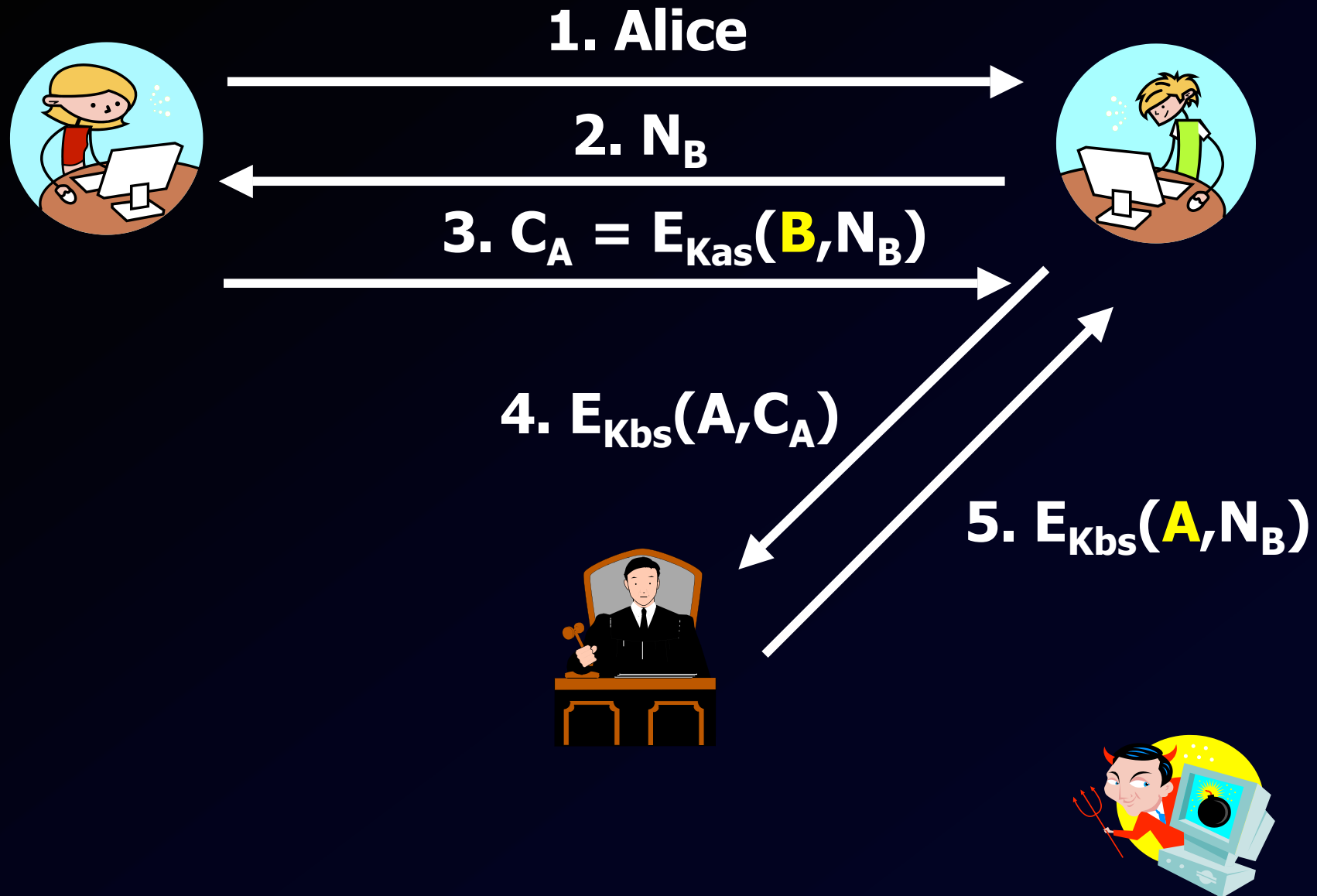
5. $E_{K_{bs}}(A, N_B)$



WOO-LAM FIX?



WOO-LAM FIX II



DENNING-SACCO

3. $CA, CB, Enc_B(TS, K_{AB}, Sign_A(TS, K_{AB}))$



1.
 A, B



2. $CA = VK_a, Sign_S(A, VK_a)$
 $CB = PK_b, Sign_S(B, PK_b)$



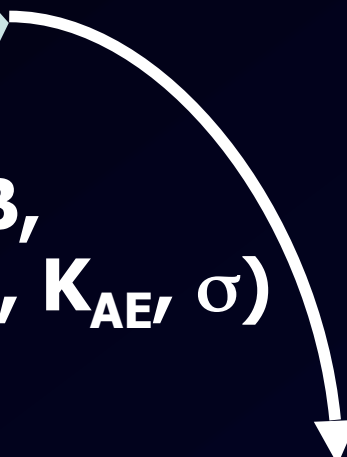
Published **1982**.

DENNING-SACCO DISASTER

3. $CA, CE, Enc_E(TS, K_{AE}, Sign_A(TS, K_{AE}))$



4. $CA, CB, Enc_B(TS, K_{AE}, \sigma)$



1. A, E



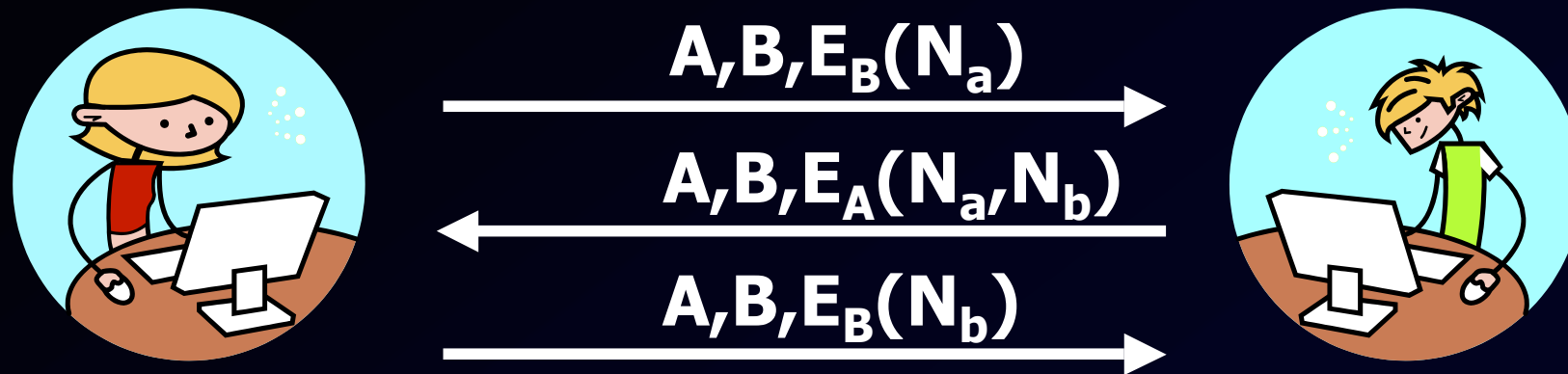
2. $CA = VK_a, Sign_S(A, VK_a)$
 $CE = PK_e, Sign_S(E, PK_e)$



Broken **1994**. 12 years later!

NEEDHAM-SCHROEDER

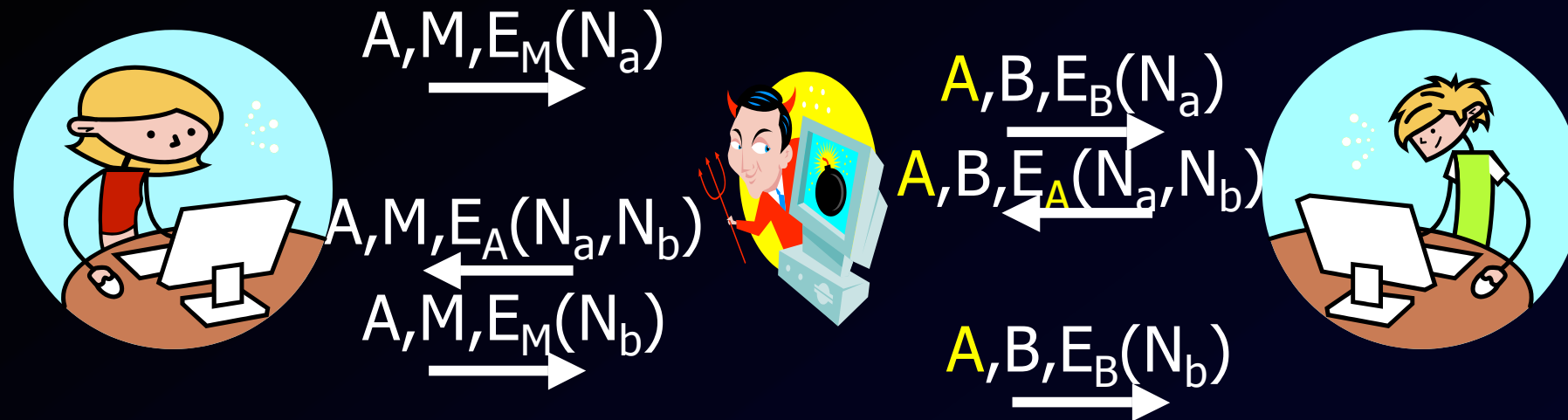
Needham & Schroeder, "Using encryption for authentication in large networks of computers," **1978**.



Shared Key: $H(N_a, N_b)$

NEEDHAM-SCHROEDER

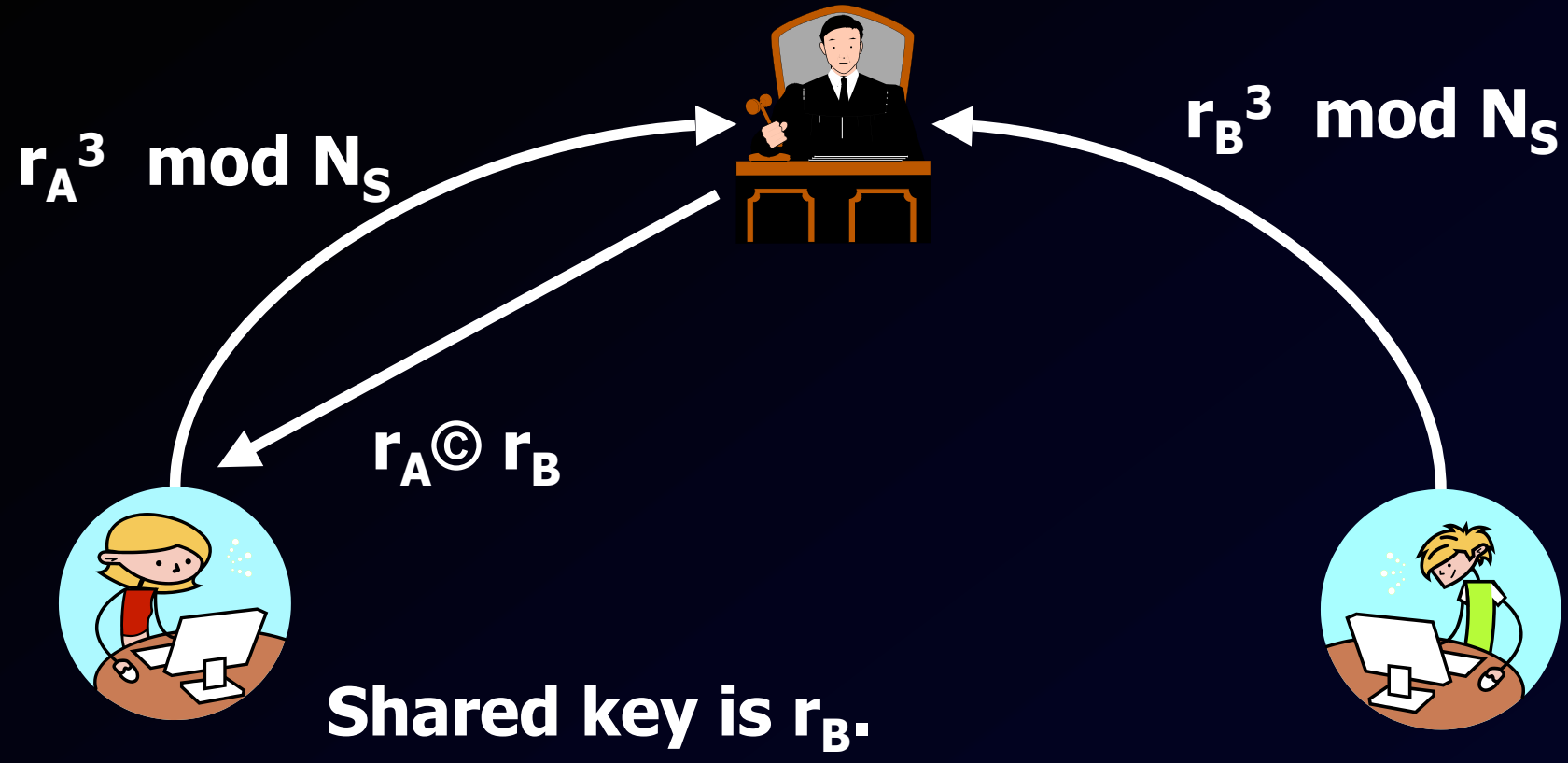
Lowe, **1995**, "Breaking and Fixing the Needham-Schroeder public key protocol"



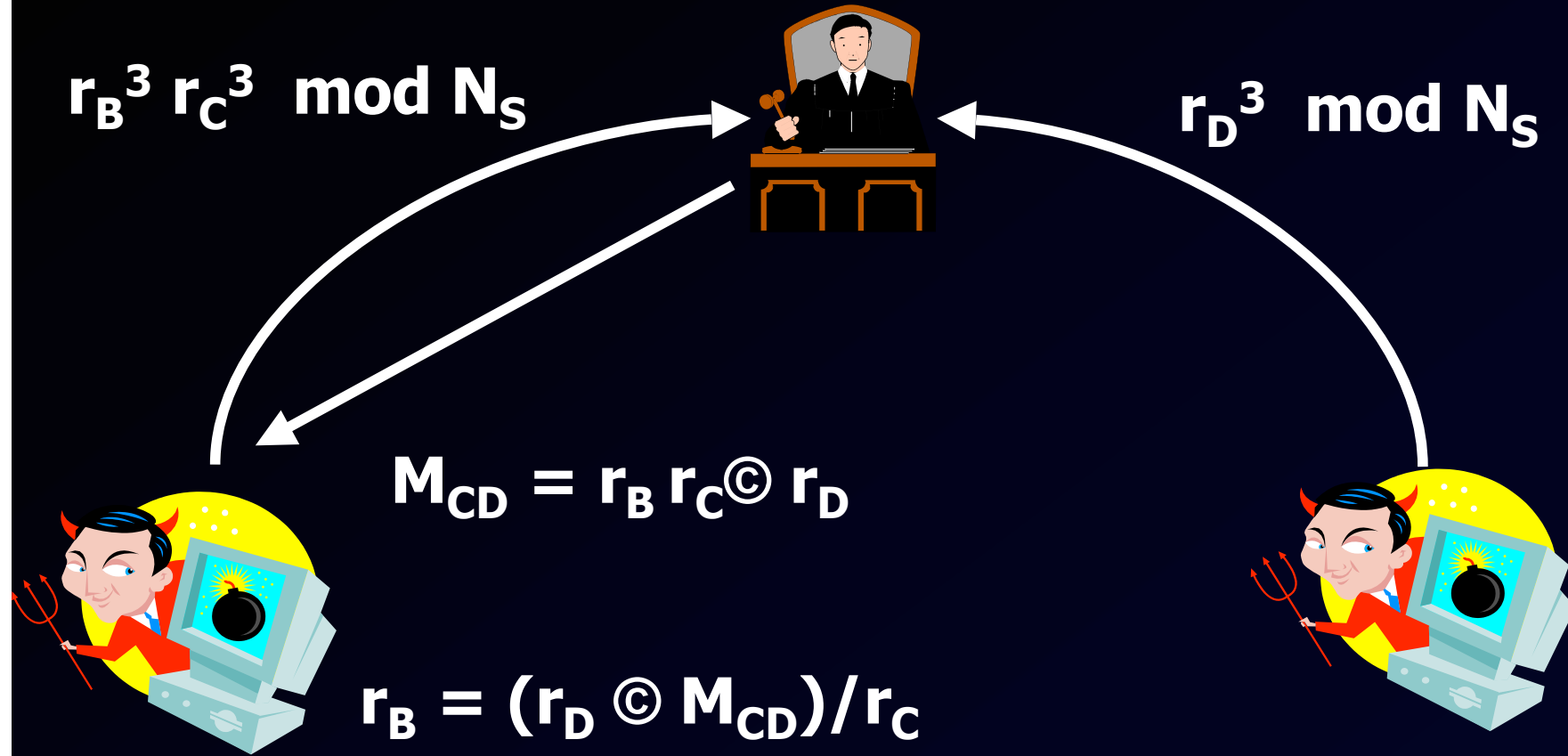
It took **17 years** to find this bug!

Shared Key: $H(N_a, N_b)$
Bob thinks it is Alice,
not Mallory!

TMN



TMN



OTHER BUGS

Many cryptosystems fail due to poor “key management” practices:

Using passwords or Hash(pw) as keys

**Choosing “random” numbers that aren’t.
(e.g. C rand(); java math.Random...)**

Reuse of passwords and keys

Unintentional “leaking” of plaintexts, keys

SIDE-CHANNEL ATTACKS

Most crypto assumes computers are “black boxes” that compute functions instantaneously.

Real computers take time to compute and interact with their environments.

Simple example: TEMPEST equipment reads monitors from their EM radiation.

TIMING ATTACKS

**Time between keystrokes depends on keys.
SSH clients send packets on keystrokes.**

www.ece.cmu.edu/~dawnsong/papers/ssh-timing.pdf

**Time to compute $x^y \bmod N$ depends on x, y .
Given many (p, c) pairs and compute times,
we can recover RSA private keys.**

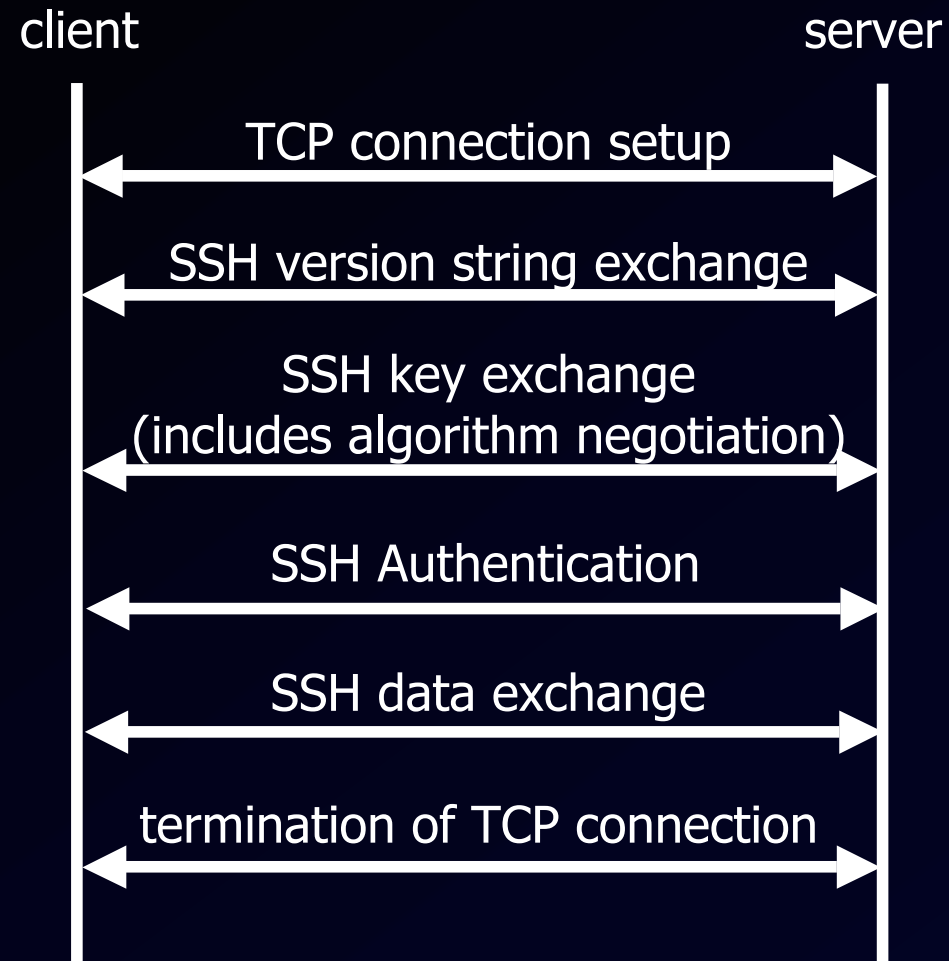
www.cryptography.com/timingattack/

**The fastest implementations of AES use four
1024-byte tables. Cache misses leak key bits.**

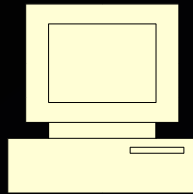
cr.ypt.to/antiforgery/cachetiming-20050414.pdf

www.wisdom.weizmann.ac.il/~tromer/papers/cache.pdf

SSH OVERVIEW

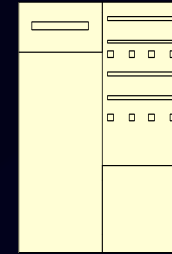


SSH KEY EXCHANGE



Client

$$X = g^x$$



Server

$$VK_S, Y, S = \text{Sign}_S(h)$$

$$Y = g^y$$

$$K = X^y = g^{xy}$$

$$K' = Y^x,$$
$$h = H(VK_S | X | Y | K')$$

use K' if $\text{Verify}(VK_S, h, S)$

$$h = H(VK_S | X | Y | K)$$

SSH AUTHENTICATION

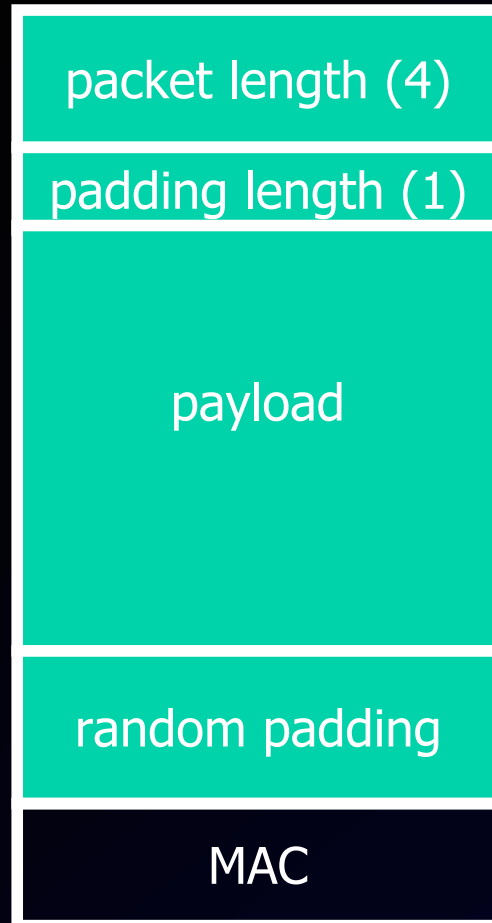
The **server** is authenticated by S and VK_S

- Client has a local database (`ssh_known_keys`) that associates each server S with VK_S .
- On 1st exchange, VK_S is added to `ssh_known_keys`.
- May also use PKI, certificates

The **client** may be authenticated in several ways:

- Password: entered by user and sent over encrypted connection.
- Public Key: the user signs a challenge from the server to be verified using her public key
- Trusted Host: the user's host signs a challenge to be verified with its public key.

SSH PACKET FORMAT



– packet length:

- length of the packet not including the MAC and the packet length field

– payload:

- useful contents of the packet
- May be compressed
- max payload size is 32768

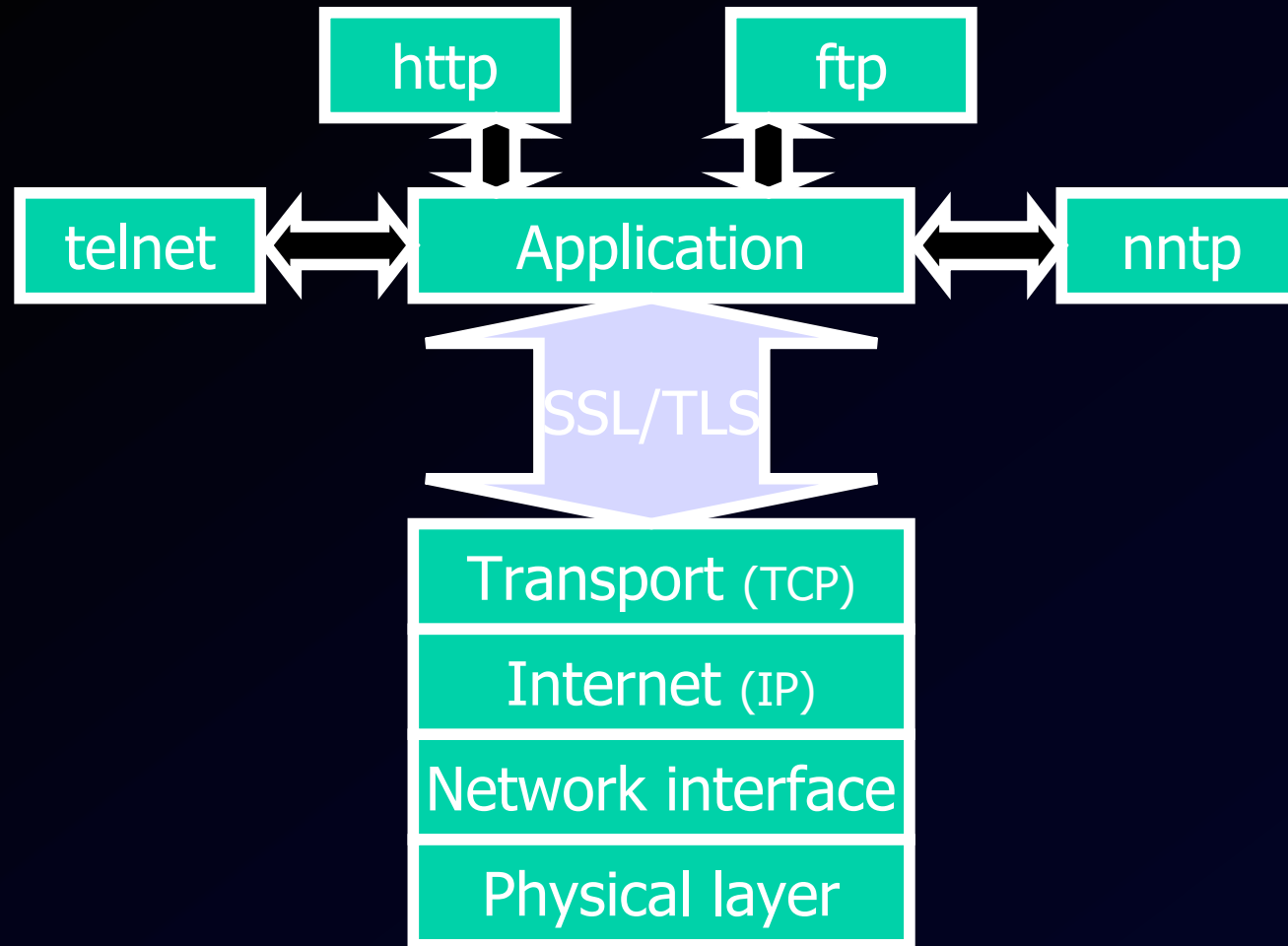
– random padding:

- 4 – 255 bytes
- total length of packet must be multiple of 16

– MAC:

- computed over **clear packet** and sequence number

TLS PROTOCOL LAYER



KEY SSL IDEAS

- **Authentication is handled using certificate authorities (PKI)**
 - CAs have “widely known” verification keys, e.g. Verisign, AT&T, MCI, Keywitness Corp Canada
 - CA supplies a signed certificate with site’s public key
- **Session integrity based on MAC of history**
 - e.g. Client & server communicate as follows:



- **Client and server compare MACs of all messages**
 - Server computes $\text{MAC}(\text{hi,hello,howareyou?})$ locally
 - Client sends MAC value under encryption
- **If intervention is detected, the session is aborted.**

IPSEC : IP-LEVEL SECURITY

IPsec provides standards to encrypt and authenticate traffic at the IP level. It has three primary security functions:

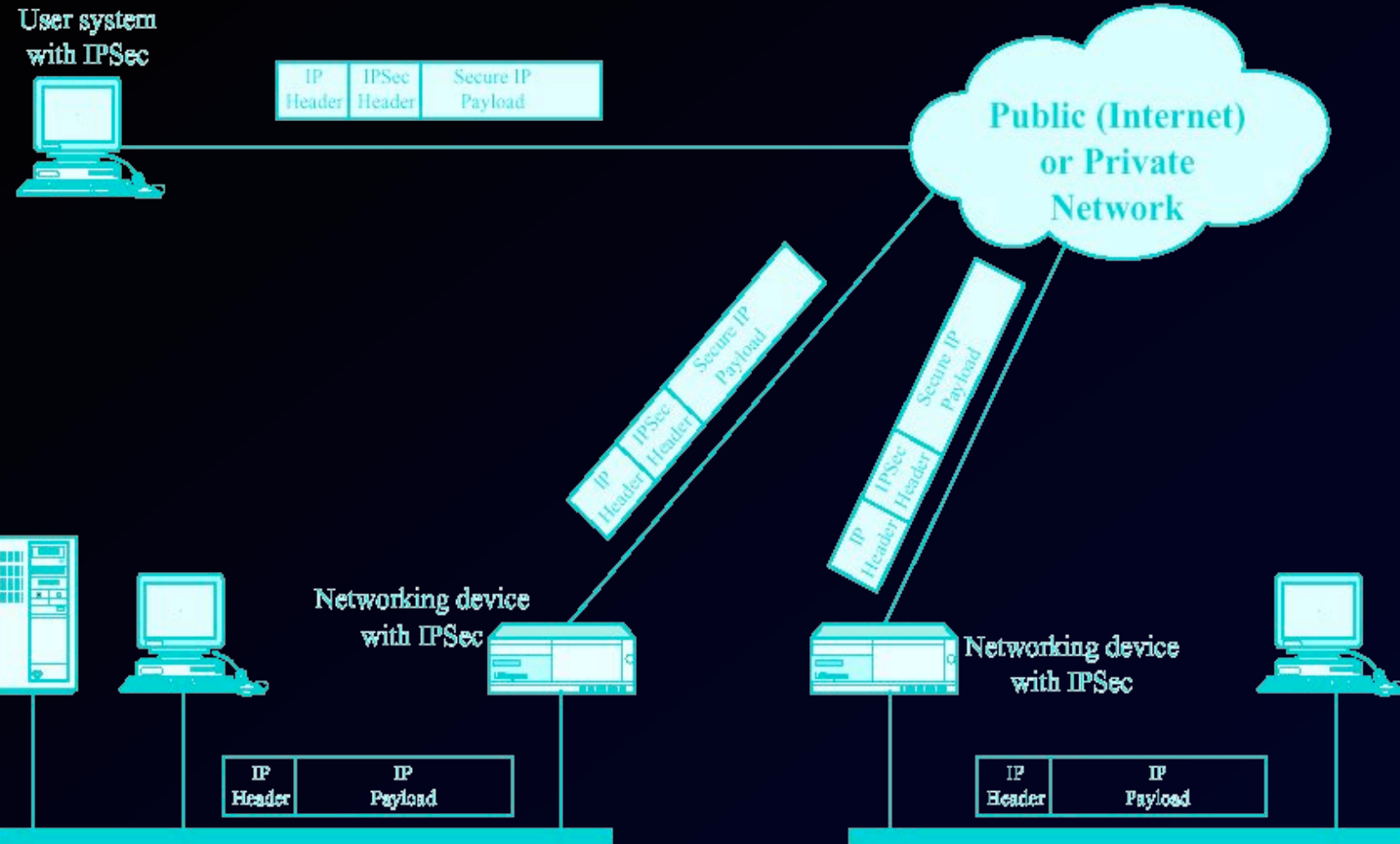
- Authentication, via “**Authentication Headers**” (AH)
- Confidentiality, via “**Encapsulated Security Payloads**” (ESP)
- Key management via **Internet Key Exchange**. (IKE)

As a network layer protocol IPsec offers several advantages over application layer protocols:

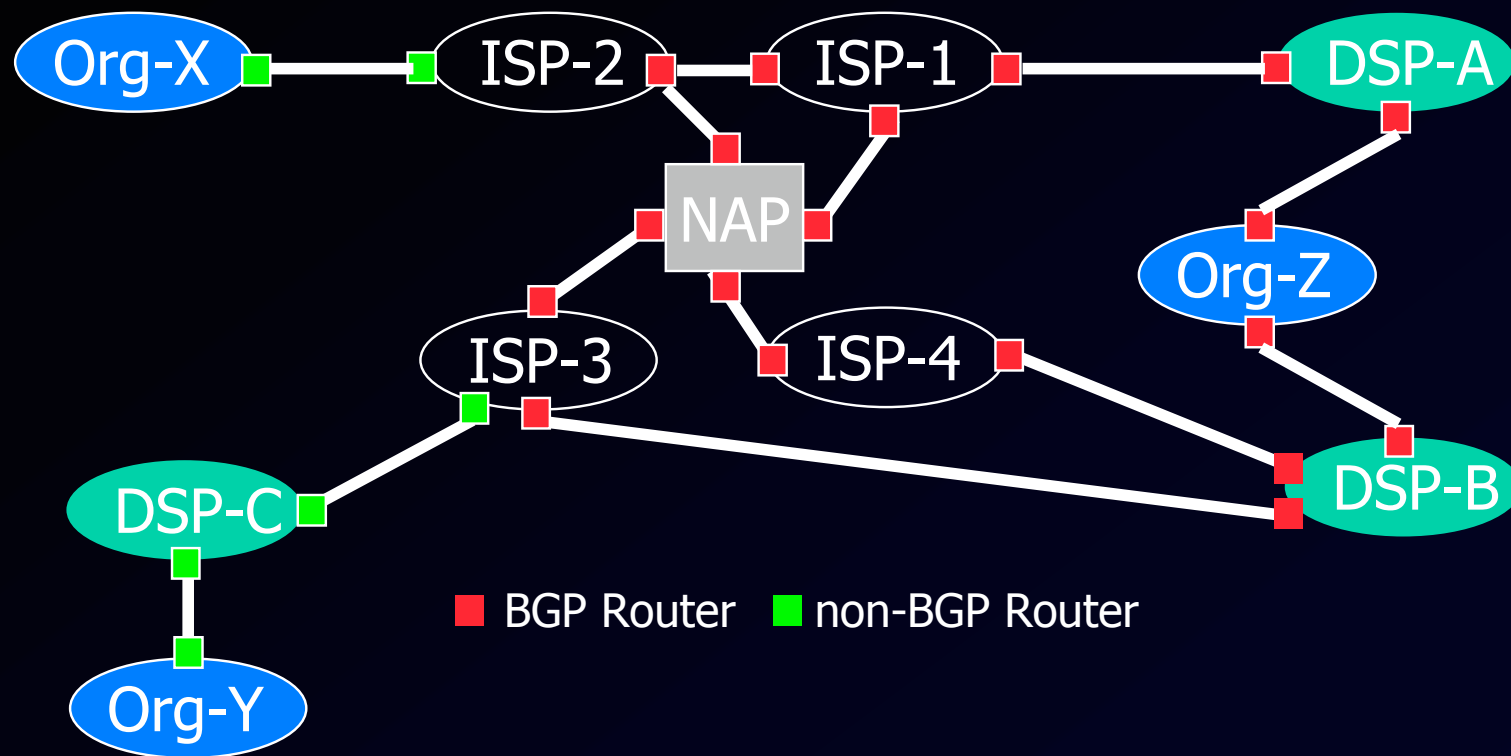
- IPsec is implemented at a **gateway**, not necessarily the desktop
- It is transparent to application programs and users (except when it's not!)

IPsec is based on **Security Associations**, a pair of hosts plus algorithms, parameters, and keys.

IPSEC USAGE SCENARIOS



BGP REVIEW



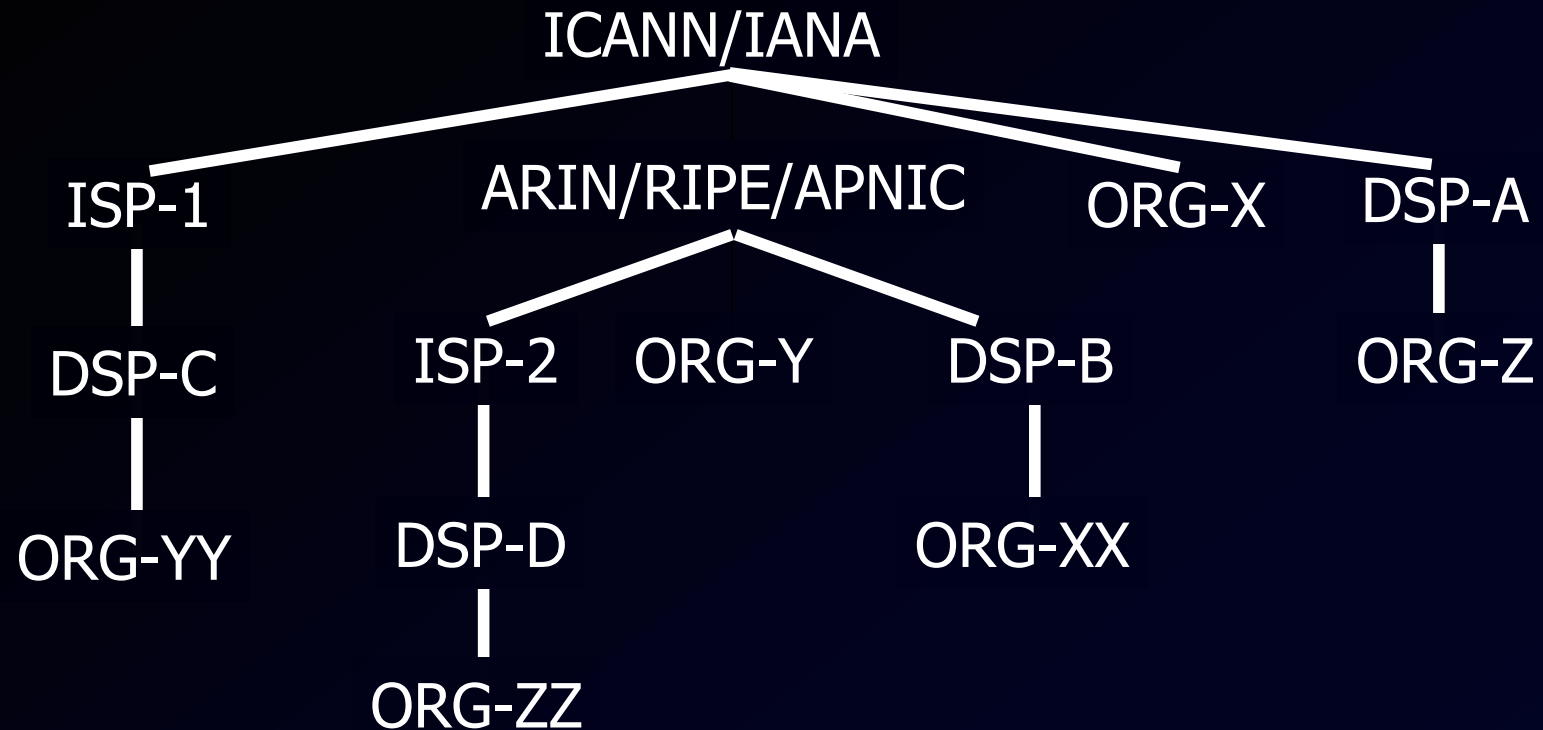
- BGP "speakers" advertise AS paths to address blocks.
- UPDATEs are generated in response to loss of connectivity or receipt of an UPDATE from a peer router

BGP SECURITY REQUIREMENTS

- **Address space “ownership” verification:**
Prove that I own IP 0.1.2.3
- **Autonomous System (AS) authentication:**
Prove that I am AS 7.
- **Router authentication and authorization:**
Prove that router 3.14.15.9 belongs to AS 7
- **Route/address advertisement authorization:**
Prove that AS 7 has a path to 42..*.**
- **Route withdrawal authorization**
- **Integrity and authenticity of BGP traffic**

SBGP PKI

Current (hierarchical) address allocation procedure:



Key idea: current organizations are *already trusted*:
PKI based on this structure assumes no additional trust.

SBGP ATTESTATIONS

SBGP uses special certificates called **attestations** to certify correctness of routes.

- **Address Attestations** are to prove that a destination address is being originated by an authorized AS. They are based on signatures and certificates from the PKI
- **Route Attestations** prove that an AS is authorized to use an AS Path. The basic idea is that a router “signs the path over” to the next hop.

Each UPDATE includes one or more Address Attestations and a set of Route Attestations

SBGP UPDATE PROPAGATION

seq:4321 nlri:a,b

% seq:321 nlri:a,b

\$ seq:21 nlri:a,b

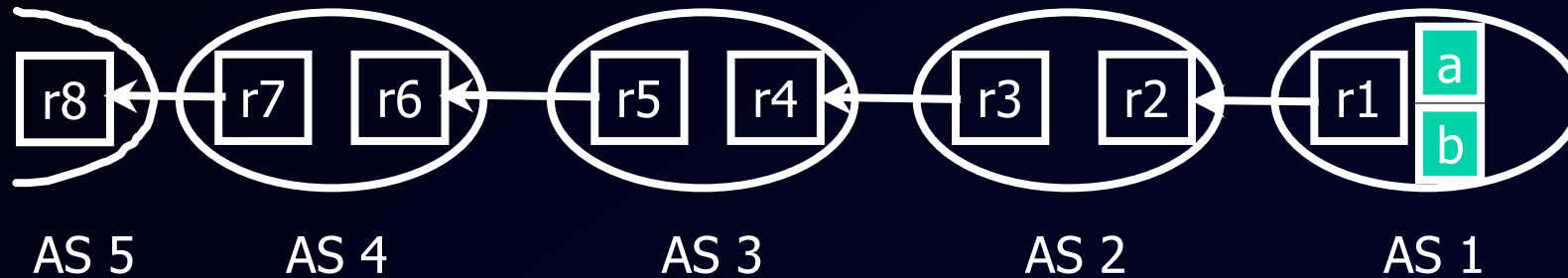
= seq:1 nlri:a,b

```
Hdr
seq:4321
RA:r7 as5 #
RA:r5 as4 %
RA:r3 as3 $
RA:as1 as2 =
AA:orga 1 a
AA:orgb 1 b
NLRI:a,b
```

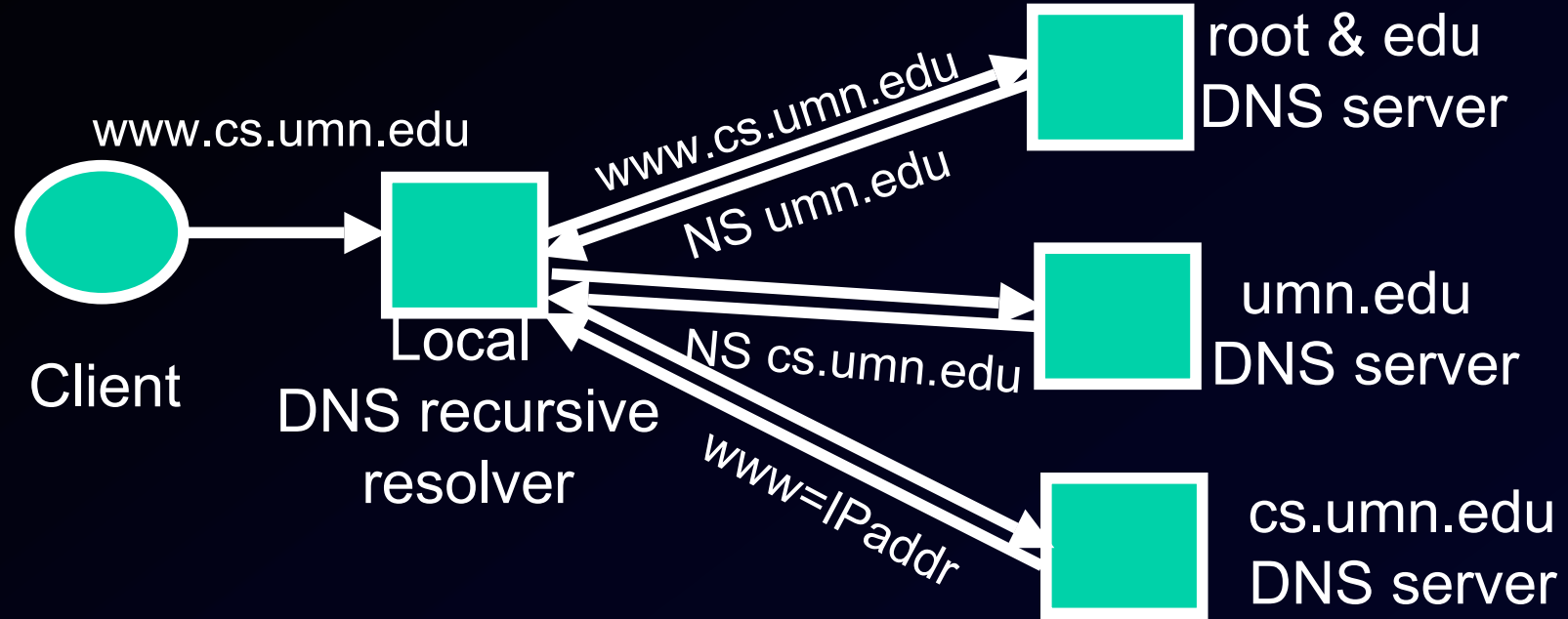
```
Hdr
seq:321
RA:r5 as4 %
RA:r3 as3 $
RA:as1 as2 =
AA:orga 1 a
AA:orgb 1 b
NLRI: a,b
```

```
Hdr
seq:21
RA:r3 as3 $
RA:as1 as2 =
AA:orga 1 a
AA:orgb 1 b
NLRI: a,b
```

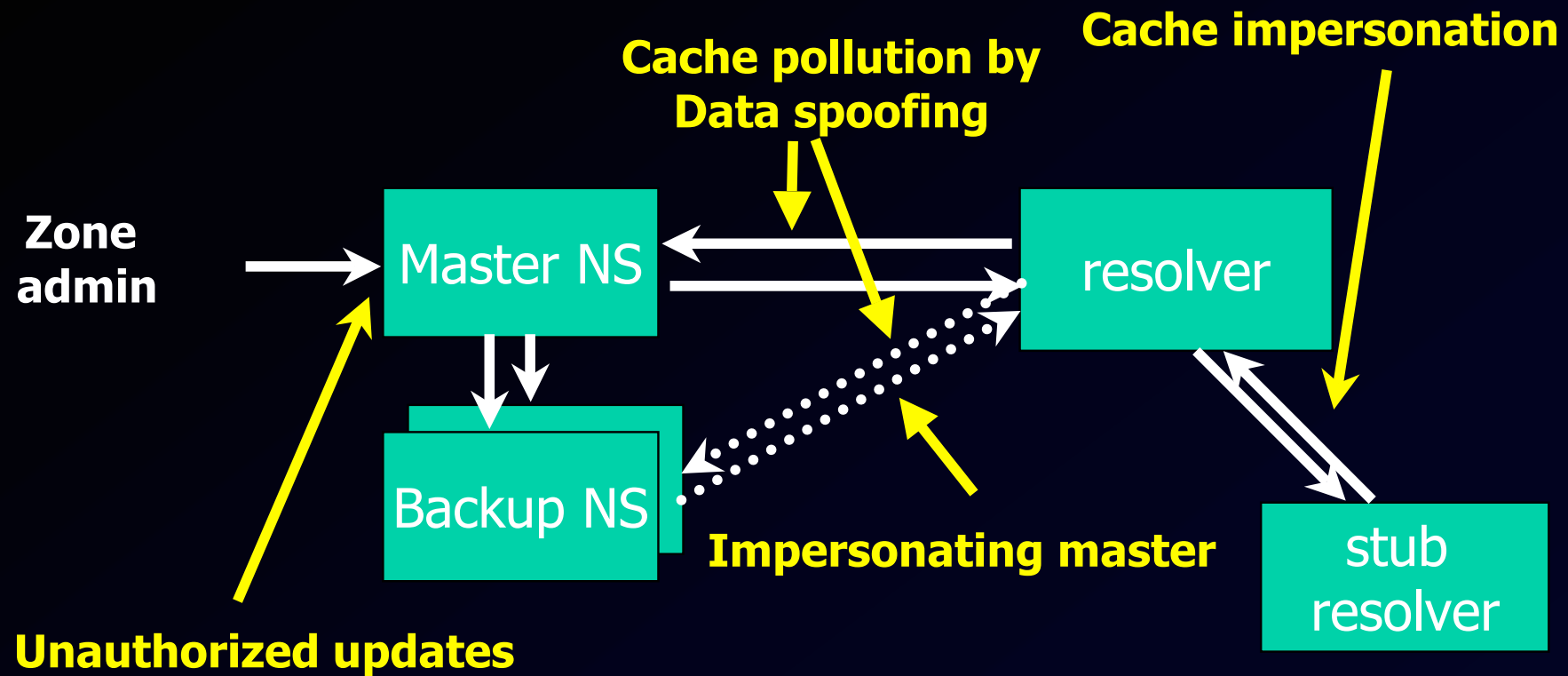
```
Hdr
seq:1
RA:as1 as2 =
AA:orga 1 a
AA:orgb 1 b
NLRI: a,b
```



DNS RESOLUTION



DNS DATA FLOW



DNSSEC

- **DNSSEC Attempts to address these issues cryptographically.**
- **In the “Ideal DNSSEC Deployment”:**
 - **There is a public key for the “root” domain.**
 - **This is used to sign certificates for the administrators of top-level domains**
 - **These admins sign certificates for subdomains**
- **DNSSEC has three new kinds of records**
 - **Signatures on results, uploaded by zone admins**
 - **Certificates, to allow verification**
 - **NSEC records, to prove negative results**

DNSSEC RESOLUTION

