

# Security 2

ELEC3227/ELEC6255

Alex Weddell  
[asw@ecs.soton.ac.uk](mailto:asw@ecs.soton.ac.uk)

# Overview

- Public Key Algorithms/RSA Encryption
- Signatures
- Message Digests
- Certificates and management of public keys
- IPsec and Firewalls
- Shared Secret Key
- Secure Naming and SSL

# Public Key Algorithms

- Definition: Encryption in which each party publishes a public part of their key and keep secret a private part of it
- Security depends on the chosen mathematical property
  - Much slower than symmetric-key, e.g., 1000x
  - So use it to set up per-session symmetric keys

# Public Key Algorithms

- Downsides of keys for symmetric-key designs:
  - Key must be secret, yet be distributed to both parties
  - For  $N$  users there are  $N^2$  pairwise keys to manage
- Public key schemes split the key into public and private parts that are mathematically related:
  - Private part is not distributed; easy to keep secret
  - Only one public key per user needs to be managed

# RSA Encryption

- RSA (by Rivest, Shamir, Adleman)
- RSA is a widely used public-key encryption method whose security is based on the difficulty of factoring large numbers
- Key generation:
  - Choose two large random primes, p and q
  - Compute the “modulus”  $n = p \times q$
  - Compute the “totient”  $z = (p - 1) \times (q - 1)$ .
  - Choose d to be  $1 < d < z$ , and **co-prime** to z (i.e. sharing no factors other than 1)
  - Find e such that  $(e \times d) \bmod z = 1$
  - Public key is  $(e, n)$ , and private key is  $(d, n)$
- Encryption (of k bit message, for numbers up to n):
  - $\text{Cipher} = \text{Plain}^e \pmod{n}$
- Decryption:
  - $\text{Plain} = \text{Cipher}^d \pmod{n}$

# RSA Encryption

- Small-scale example of RSA encryption
  - In this example, For  $p=3, q=11 \rightarrow n=33, z=20 \rightarrow d=7, e=3$

Plaintext (P)		Ciphertext (C)		After decryption	
Symbolic	Numeric	$P^3$	$P^3 \pmod{33}$	$C^7$	$C^7 \pmod{33}$
S	19	6859	28	13492928512	19
U	21	9261	21	1801088541	21
Z	26	17576	20	1280000000	26
A	01	1	1	1	01
N	14	2744	5	78125	14
N	14	2744	5	78125	14
E	05	125	26	8031810176	05

Encryption:  $C = P^3 \text{ mod } 33$

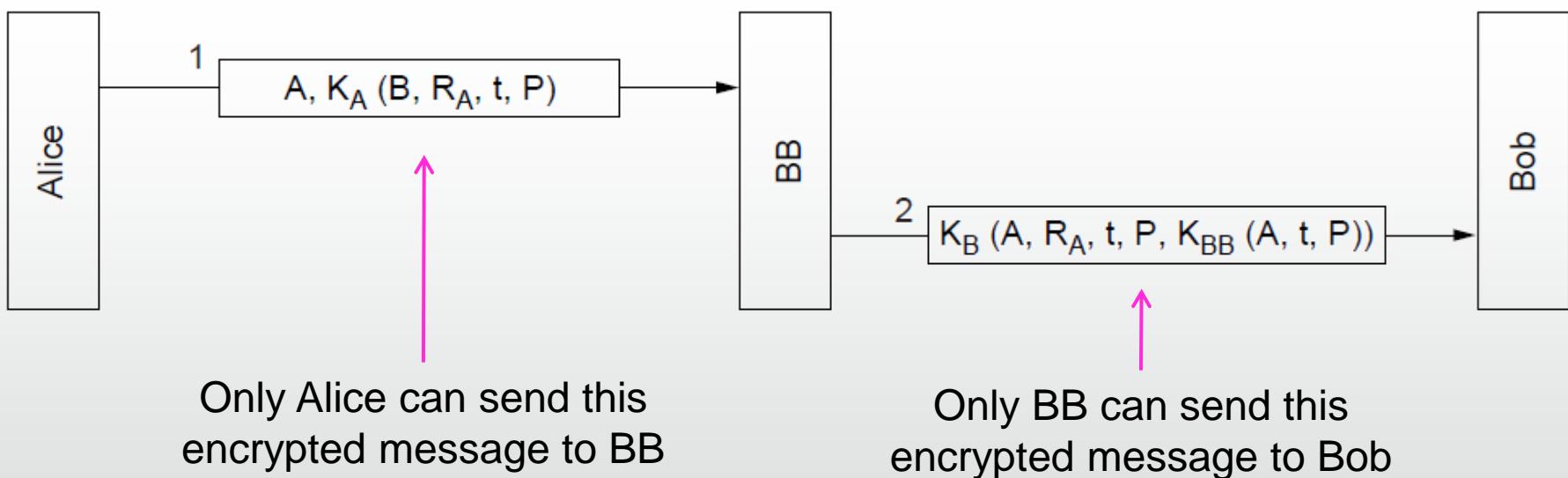
Decryption:  $P = C^7 \text{ mod } 33$

# Digital Signatures

- Let receiver verify that message is authentic
- Requirements for a signature:
  - Receiver can verify claimed identity of sender.
  - Sender cannot later repudiate contents of message.
  - Receiver cannot have concocted message himself.

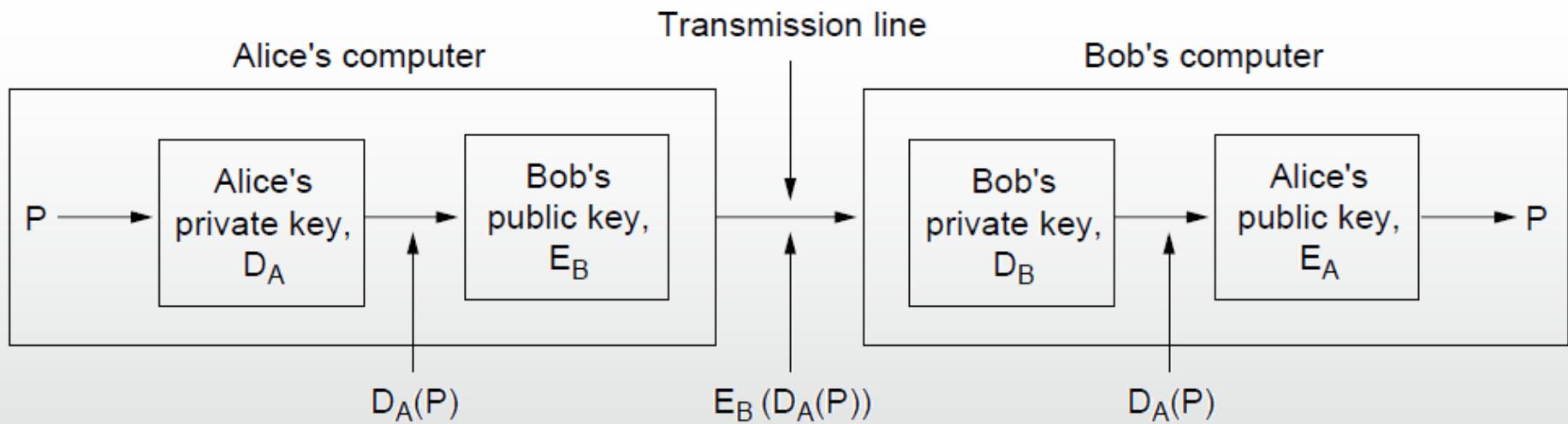
# Symmetric-Key Signatures

- Alice and Bob each trust and share a key with Big Brother; Big Brother doesn't trust anyone
  - A=Alice, B=Bob, P=message,  $R_A$ =random, t=time



# Public-Key Signatures

- No Big Brother and assumes encryption and decryption are inverses that can be applied in either order
  - But relies on private key kept and secret
  - RSA & DSS (Digital Signature Standard) widely used

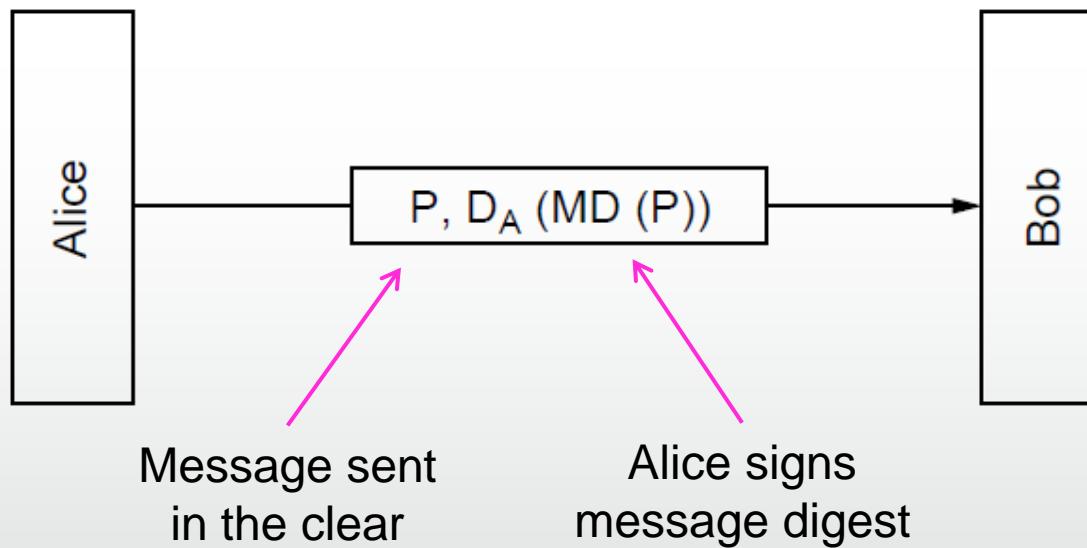


# Message Digests

- Used when *authentication* is needed, but not *secrecy*
- Message Digest (MD) converts arbitrary-size message ( $P$ ) into a fixed-size identifier  $MD(P)$  with properties:
  - Given  $P$ , easy to compute  $MD(P)$ .
  - Given  $MD(P)$ , effectively impossible to find  $P$ .
  - Given  $P$  no one can find  $P'$  so that  $MD(P') = MD(P)$ .
  - Changing 1 bit of  $P$  produces very different  $MD$ .
- Message digests (also called cryptographic hash) can “stand for” messages in protocols, e.g., authentication

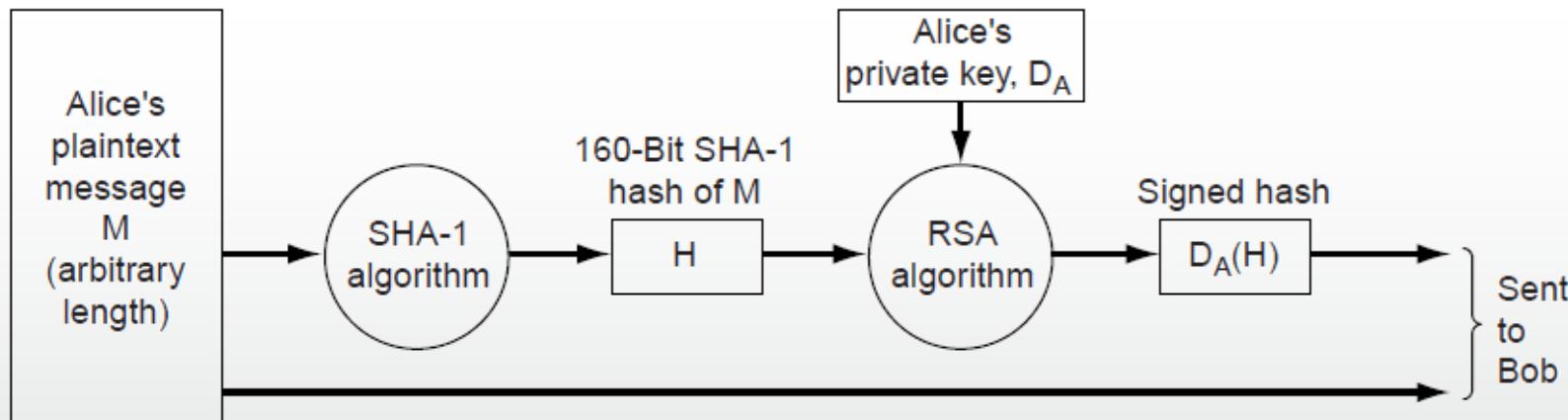
# Message Digests

- Public-key signature for message authenticity but not confidentiality with a message digest

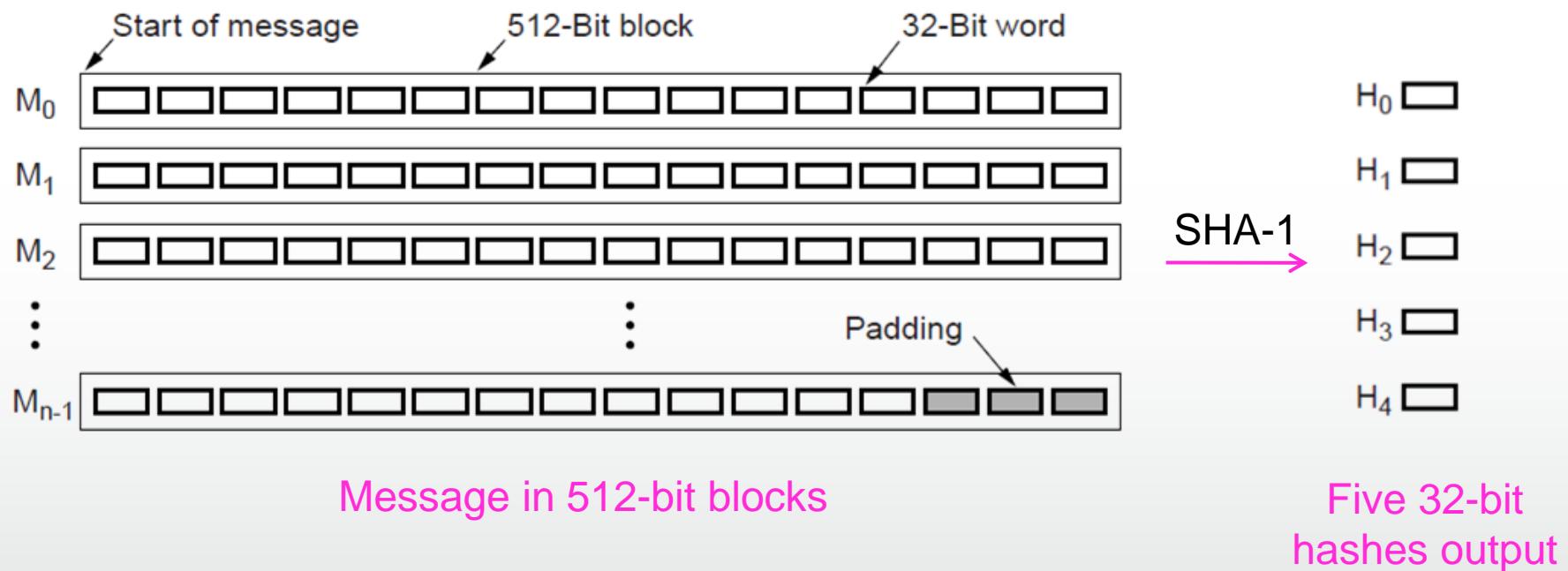


# Message Digests

- In more detail: example of using SHA-1 message digest and RSA public key for signing nonsecret messages



# Message Digests

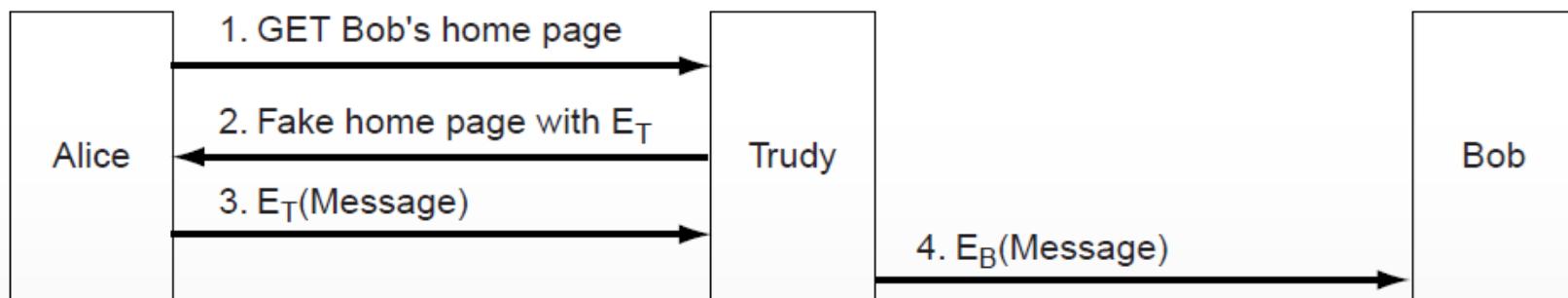


# Message Digests

- How hard is it to find a message  $P'$  that has the same message digest as  $P$ ?
  - Such a collision will allow  $P'$  to be substituted for  $P$ !
- Analysis:
  - $N$  bit hash has  $2^N$  possible values
  - Expect to test  $2^N$  messages given  $P$  to find  $P'$
  - But expect only  $2^{N/2}$  messages to find a collision
  - This is the **birthday attack**
- So it is easier than it may appear to spoof a message with this scheme.

# Management of Public Keys

- Trudy can subvert encryption if she can fake Bob's public key; Alice and Bob will not necessarily know



Trudy replaces  $E_B$  with  $E_T$  and acts as a “man in the middle”

- We need a trusted way to distribute public keys!

# Certificates

- CA (Certification Authority) issues signed statements about public keys; users trust CA and it can be offline

I hereby certify that the public key  
19836A8B03030CF83737E3837837FC3s87092827262643FFA82710382828282A  
belongs to  
Robert John Smith  
12345 University Avenue  
Berkeley, CA 94702  
Birthday: July 4, 1958  
Email: bob@superduper.net.com

SHA-1 hash of the above certificate signed with the CA's private key

A possible certificate

# X.509 and SSL

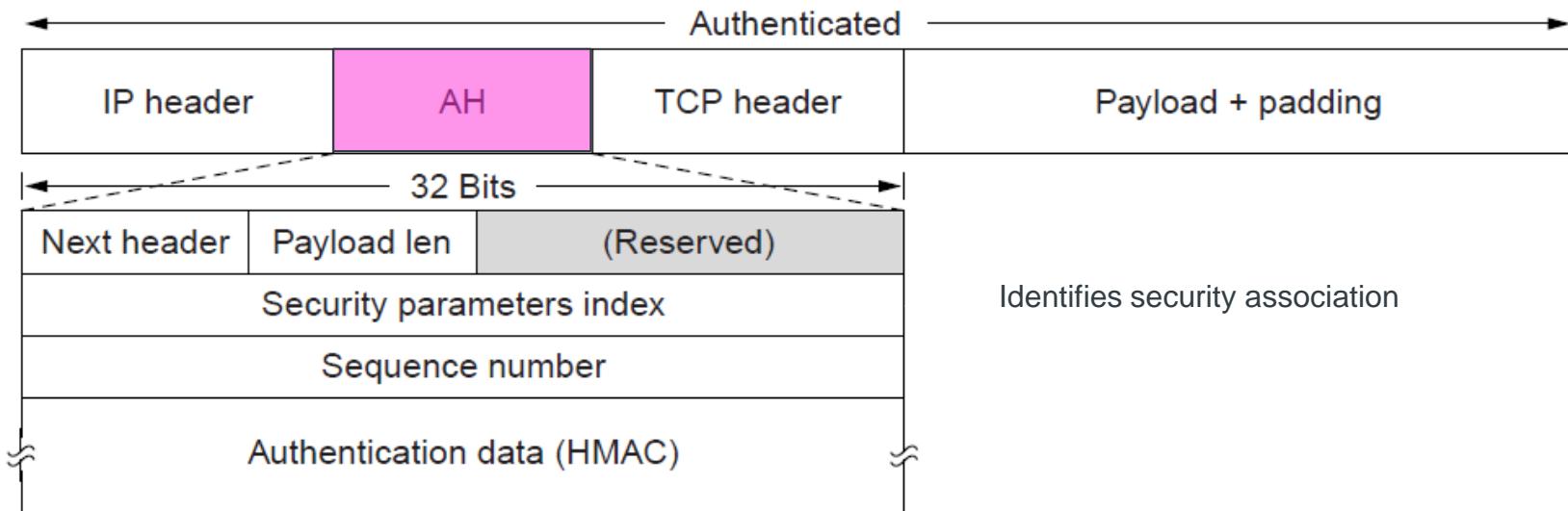
- X.509 is the standard for widely used certificates – e.g. SSL for Web browsing

Field	Meaning
Version	Which version of X.509
Serial number	This number plus the CA's name uniquely identifies the certificate
Signature algorithm	The algorithm used to sign the certificate
Issuer	X.500 name of the CA
Validity period	The starting and ending times of the validity period
Subject name	The entity whose key is being certified
Public key	The subject's public key and the ID of the algorithm using it
Issuer ID	An optional ID uniquely identifying the certificate's issuer
Subject ID	An optional ID uniquely identifying the certificate's subject
Extensions	Many extensions have been defined
Signature	The certificate's signature (signed by the CA's private key)

Basic fields in X.509 certificates

# IPsec

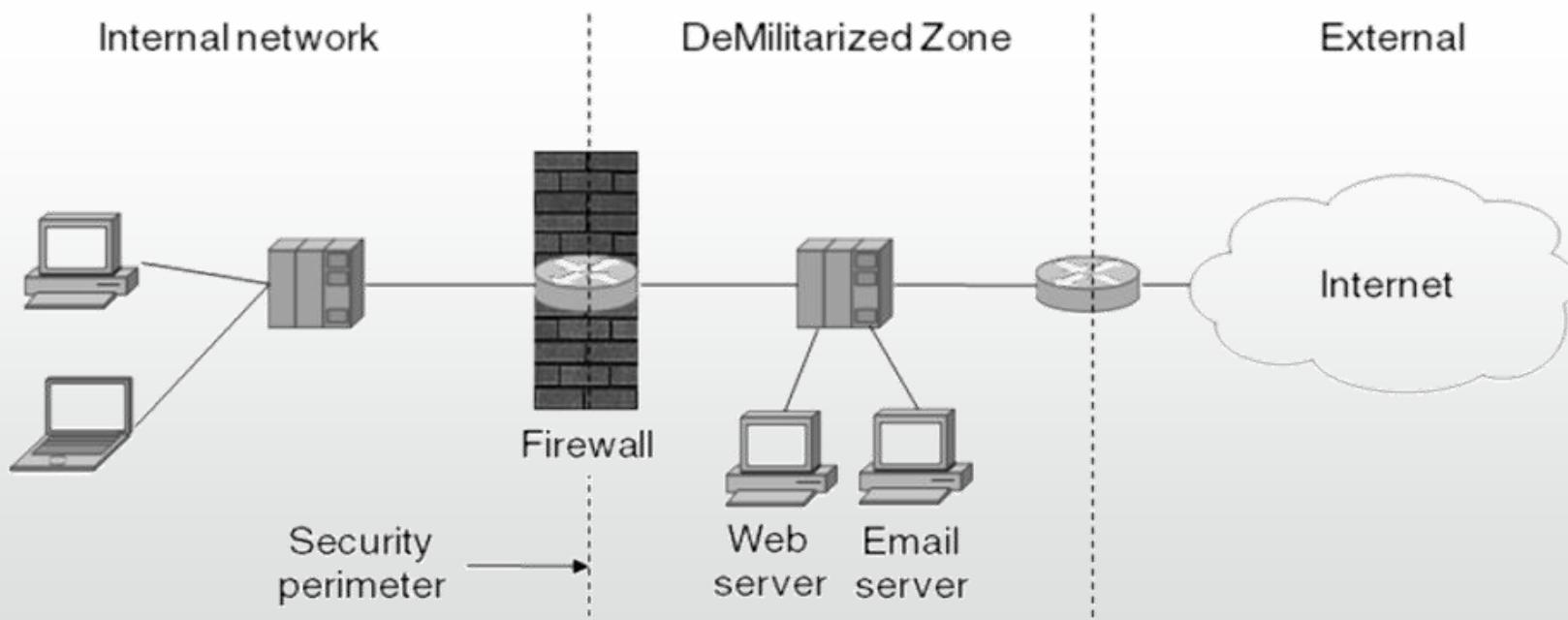
- Original intention was that security would only go in Application layer
- However, this means all applications need to be security-aware
- Add scheme to transport layer – end-to-end but no need to change application
  - Secret keys are set up for packets between endpoints called security associations
  - Adds AH header; inserted after IP in transport mode



AH (Authentication Header) provides integrity and anti-replay

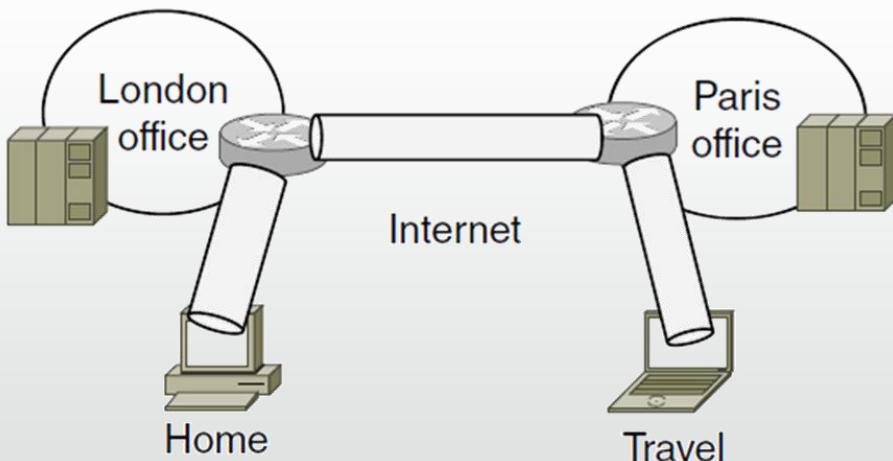
# Firewalls

- A firewall protect an internal network by filtering packets
  - Can have stateful rules about what packets to pass
    - e.g., no incoming packets to port 80 (Web) or 25 (SMTP)
  - DMZ helps to separate internal from external traffic
    - e.g., run Web and Email servers there

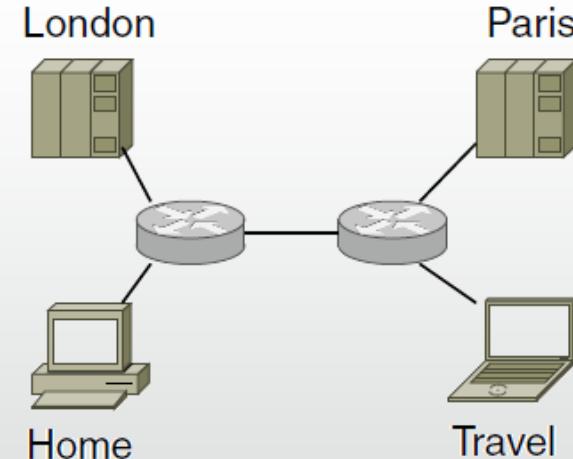


# Virtual Private Networks

- VPNs (Virtual Private Networks) join disconnected islands of a logical network into a single virtual network
  - Islands are joined by tunnels over the Internet
- VPN traffic travels over the Internet but hosts separated from the Internet
  - Need a gateway to send traffic in/out of VPN



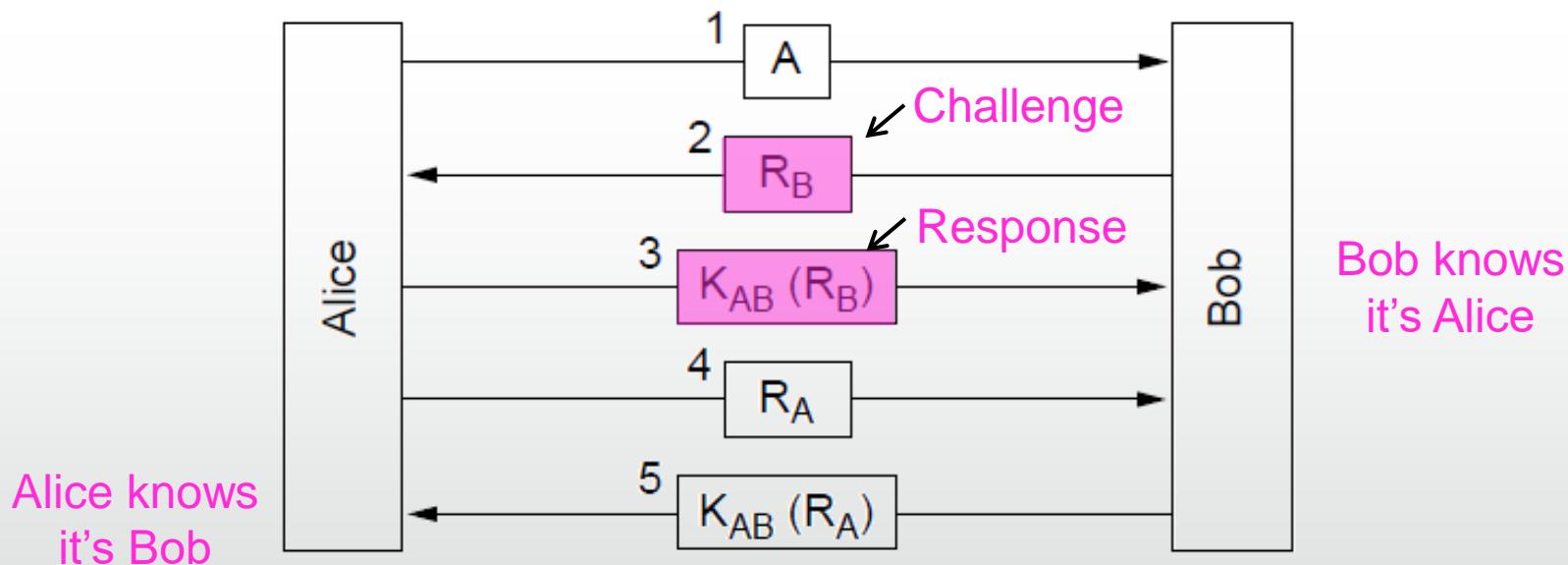
VPN joining London, Paris, Home, and Travel



Topology as seen from inside the VPN

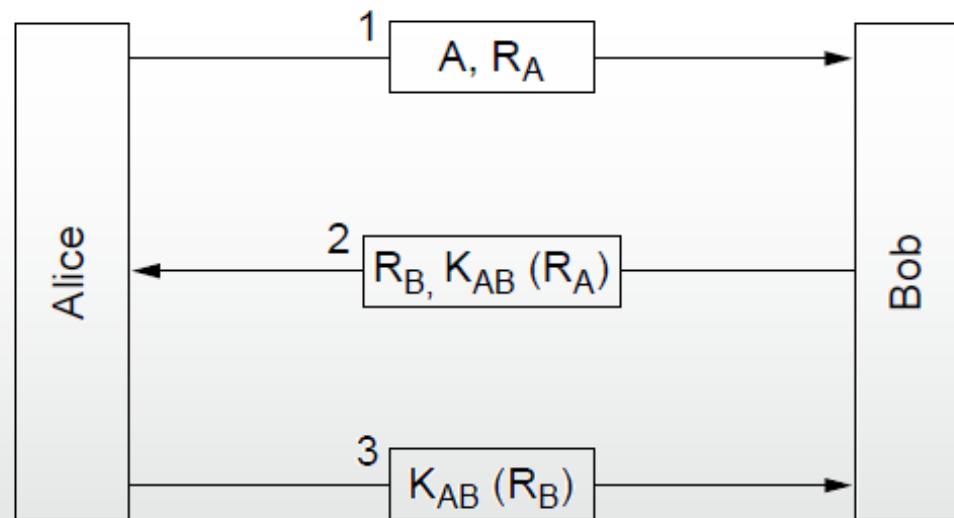
# Shared Secret Key

- Authenticating with a challenge-response (first attempt)
  - Alice (A) and Bob (B) share a key  $K_{AB}$
  - $R_X$  is random,  $K_X(M)$  is  $M$  encrypted with key  $K_X$



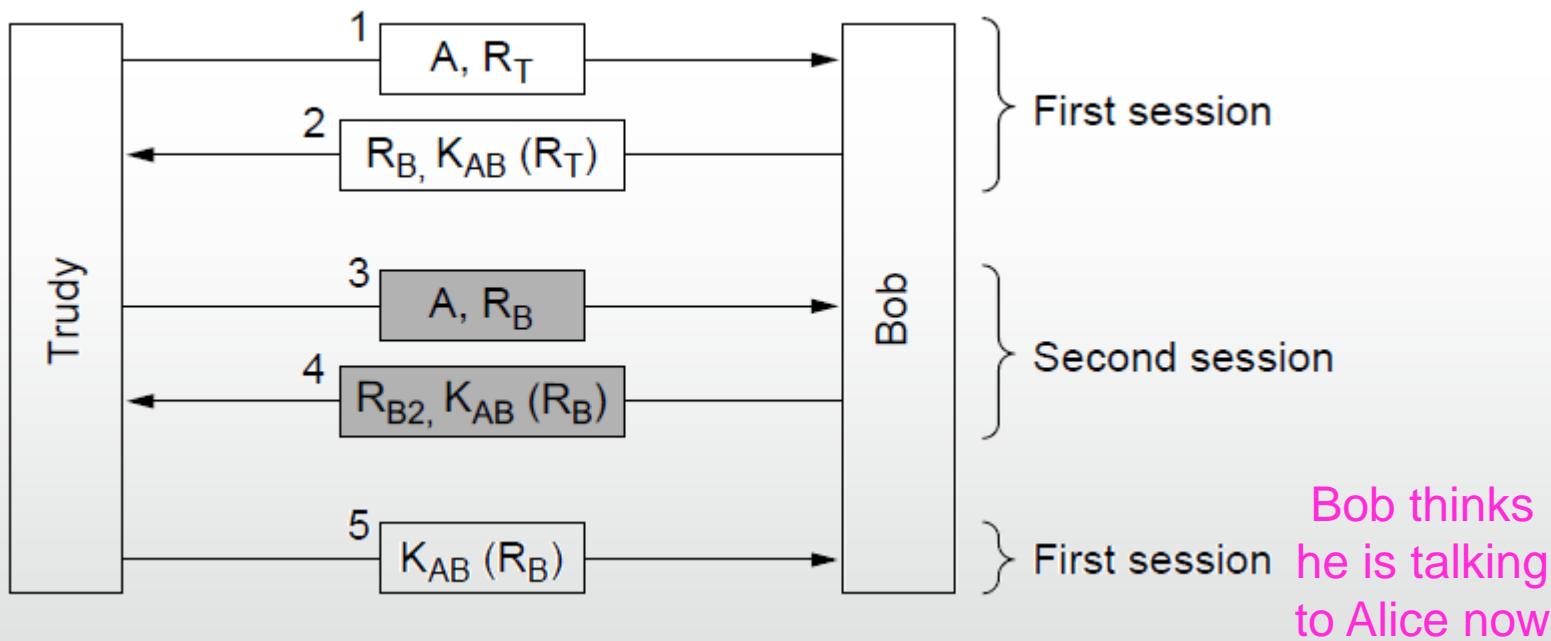
# Shared Secret Key

- A shortened two-way authentication (second attempt)
  - But it is vulnerable to reflection attack



# Shared Secret Key

- Trudy impersonates Alice to Bob with reflection attack
  - Second session gets Bob to give Trudy the response

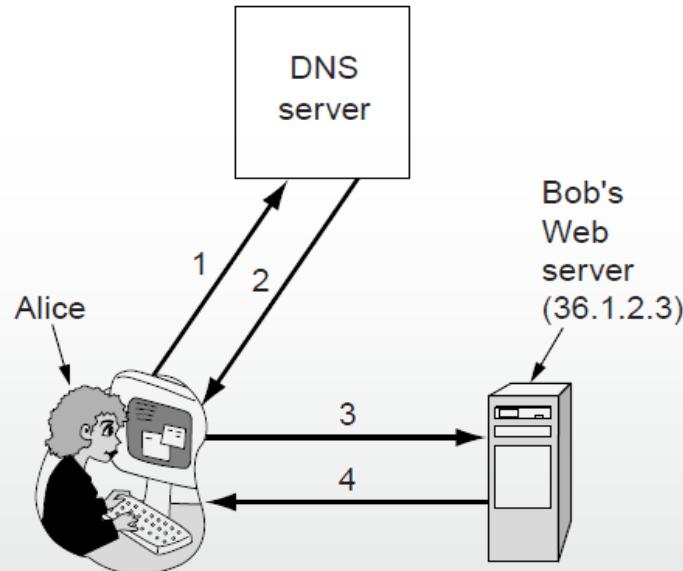


# Shared Secret Key

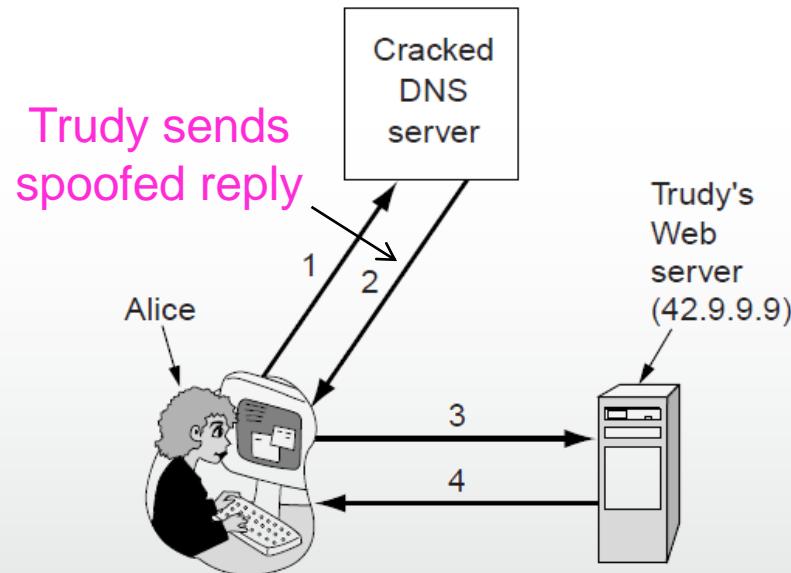
- Moral: *Designing a correct authentication protocol is harder than it looks; errors are often subtle.*
- General design rules for authentication:
  1. Have initiator prove who she is before responder
  2. Initiator, responder use different keys
  3. Draw challenges from different sets
  4. Make protocol resistant to attacks involving second parallel session

# Secure Naming

- DNS names are included as part of URLs – so spoofing DNS resolution causes Alice contact Trudy not Bob



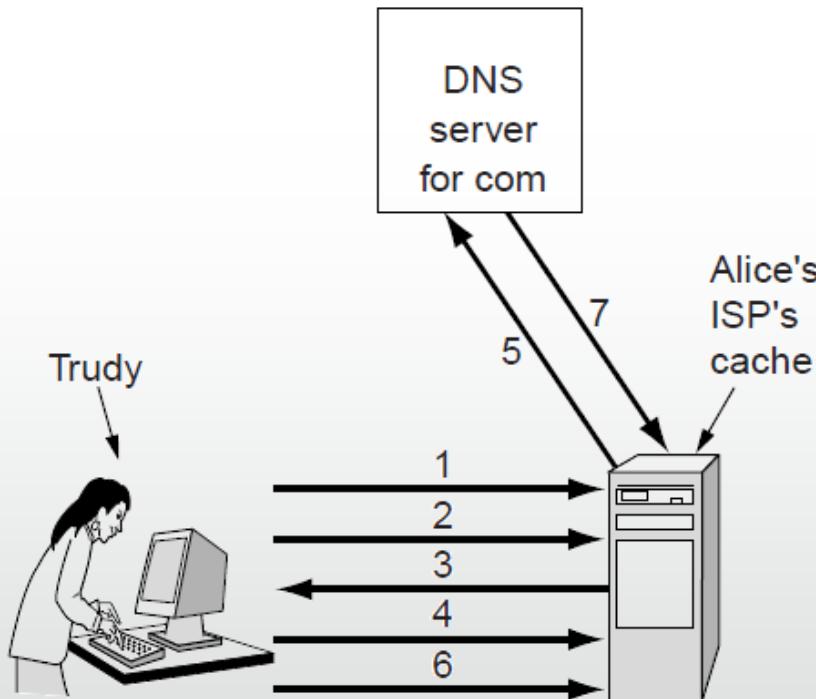
1. Give me Bob's IP address
2. 36.1.2.3 (Bob's IP address)
3. GET index.html
4. Bob's home page



1. Give me Bob's IP address
2. 42.9.9.9 (Trudy's IP address)
3. GET index.html
4. Trudy's fake of Bob's home page

# Secure Naming

- How Trudy spoofs the DNS for *bob.com* in more detail
  - To counter, DNS servers randomize seq. numbers



1. Look up `foobar.trudy-the-intruder.com`  
(to force it into the ISP's cache)
2. Look up `www.trudy-the-intruder.com`  
(to get the ISP's next sequence number)
3. Request for `www.trudy-the-intruder.com`  
(Carrying the ISP's next sequence number, n)
4. Quick like a bunny, look up `bob.com`  
(to force the ISP to query the com server in step 5)
5. Legitimate query for `bob.com` with `seq = n+1`
6. Trudy's forged answer: Bob is 42.9.9.9, `seq = n+1`
7. Real answer (rejected, too late)

# Secure Naming

- DNSsec (DNS security) adds strong authenticity to DNS
  - Responses are signed with public keys
  - Public keys are included; client starts with top-level
  - Also optional anti-spoofing to tie request/response

Domain name	Time to live	Class	Type	Value
bob.com.	86400	IN	A	36.1.2.3
bob.com.	86400	IN	KEY	3682793A7B73F731029CE2737D...
bob.com.	86400	IN	SIG	86947503A8B848F5272E53930C...

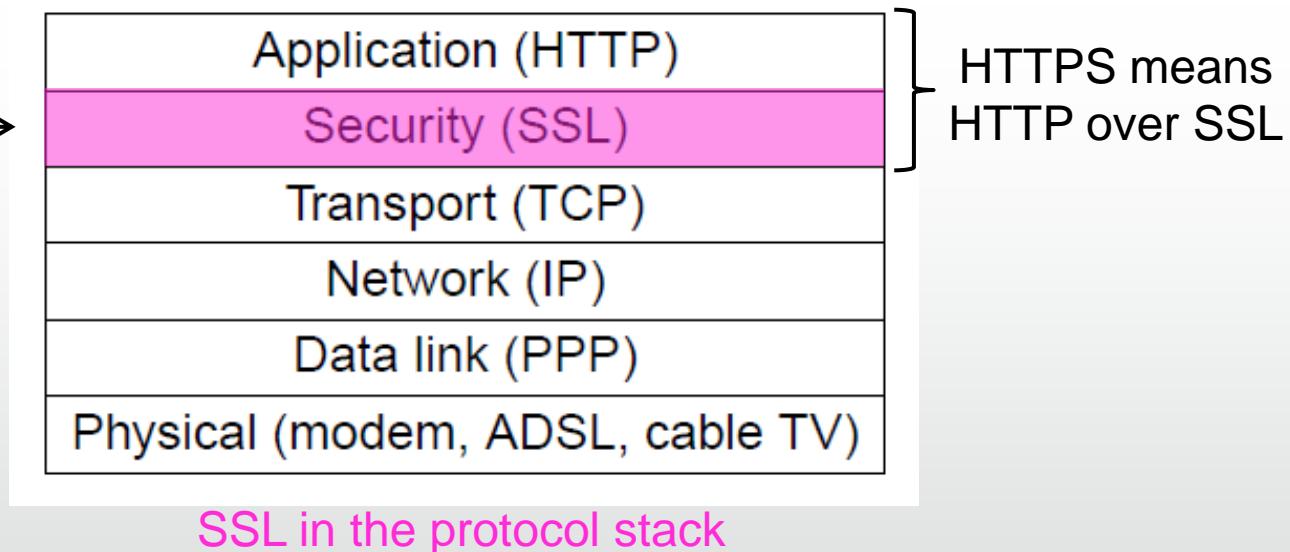
Resource Record set for *bob.com*.

Has Bob's public key (KEY), and is signed by .com server (SIG)

# SSL – Secure Sockets Layer

- SSL provides an authenticated, secret connection between two sockets; uses public keys with X.509
  - TLS (Transport Layer Security) is the IETF version

SSL runs on  
top of TCP  
and below  
the application



# Summary

- Public Key Algorithms/RSA Encryption
- Signatures
- Message Digests
- Certificates and management of public keys
- IPsec and Firewalls
- Shared Secret Key
- Secure Naming and SSL