Securing Fixed and Wireless Networks, COMP4337/9337
WK02-02Authenticaton, Key Distribution (Asymmetric)

Never Stand Still

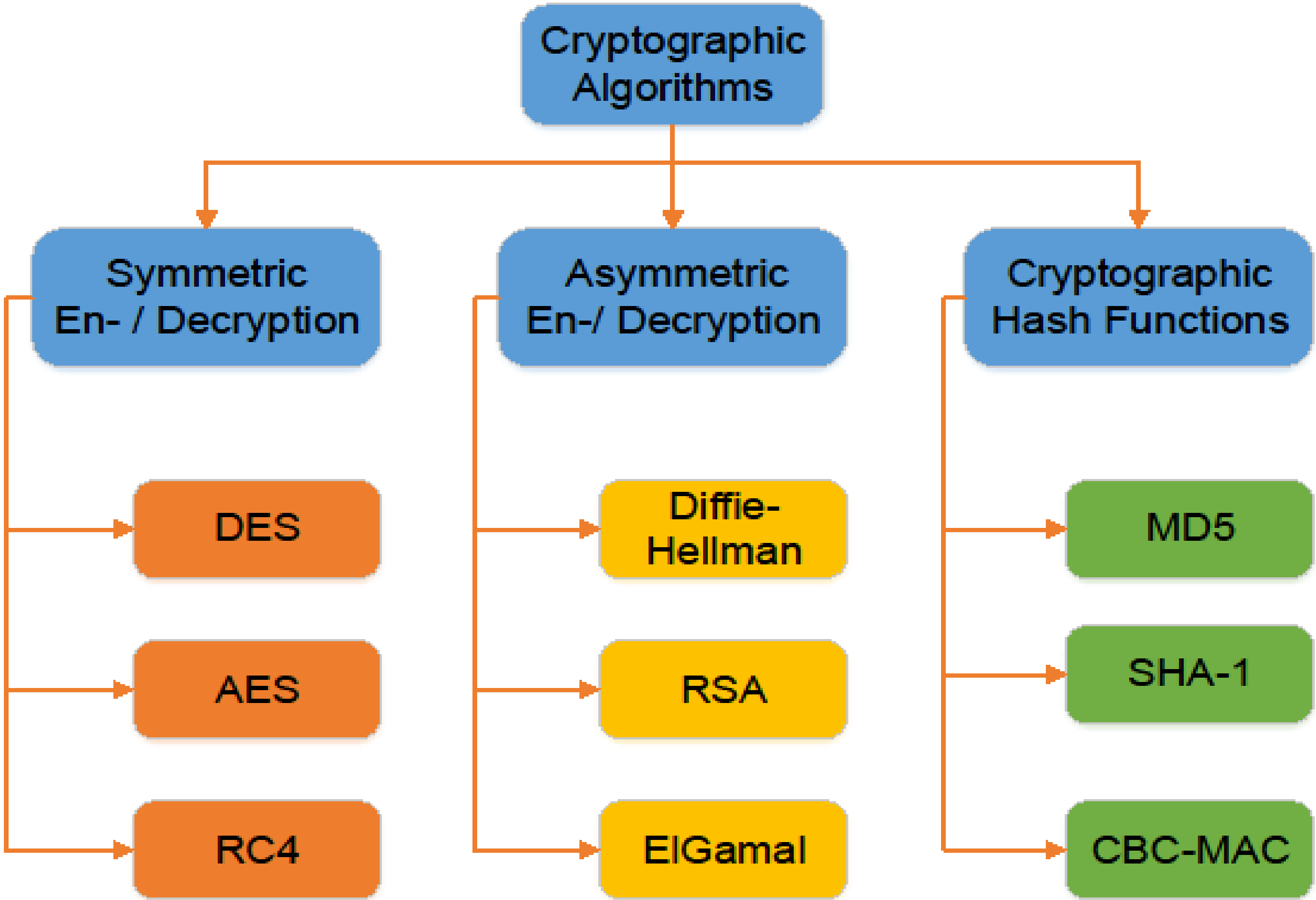
Professor Sanjay K. Jha

School of Computer Science and Engineering, UNSW

Today's Agenda

- Authentication Recap
- Key distribution using asymmetric encryption
  - Public-key distribution of secret keys
- Formal Method for Protocol Specification and Verification: AVISPA Tool

Recap



Cryptographic Algorithms
Symmetric En- / Decryption
Asymmetric En-/ Decryption
Cryptographic Hash Functions
DES
AES
RC4
Diffie-Hellman
RSA
ElGamal
MD5
SHA-1
CBC-MAC

Recap Authentication Basics

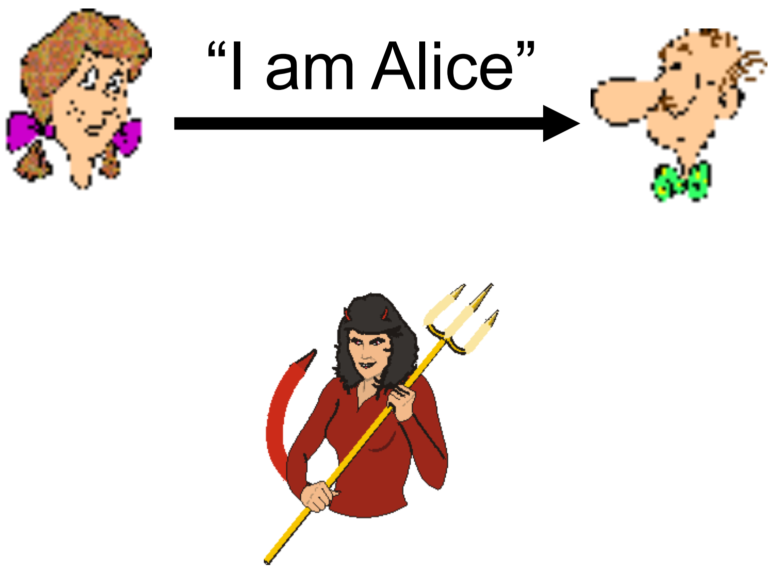
- Quick recap, possibly already done in 3331/9331 (Kurose-Ross Ch8)

- These are basic building blocks
  - Make sure you understand this well as they help material covered in this subject.

Authentication

Goal: Bob wants Alice to “prove” her identity to him

Protocol ap1.0: Alice says “I am Alice”

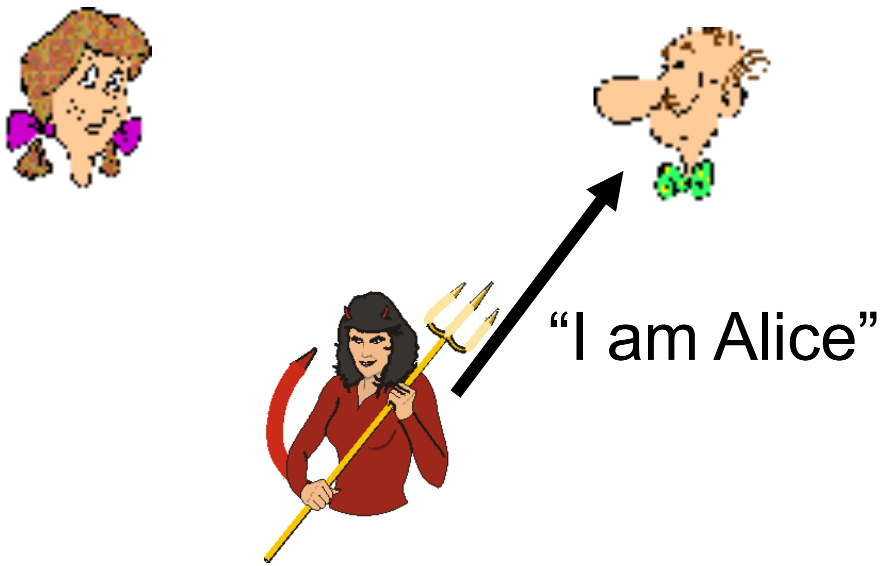


Failure scenario??

Authentication

Goal: Bob wants Alice to “prove” her identity to him

Protocol ap1.0: Alice says “I am Alice”



In a network,
Bob can not “see” Alice, so
Eve simply declares
herself to be Alice

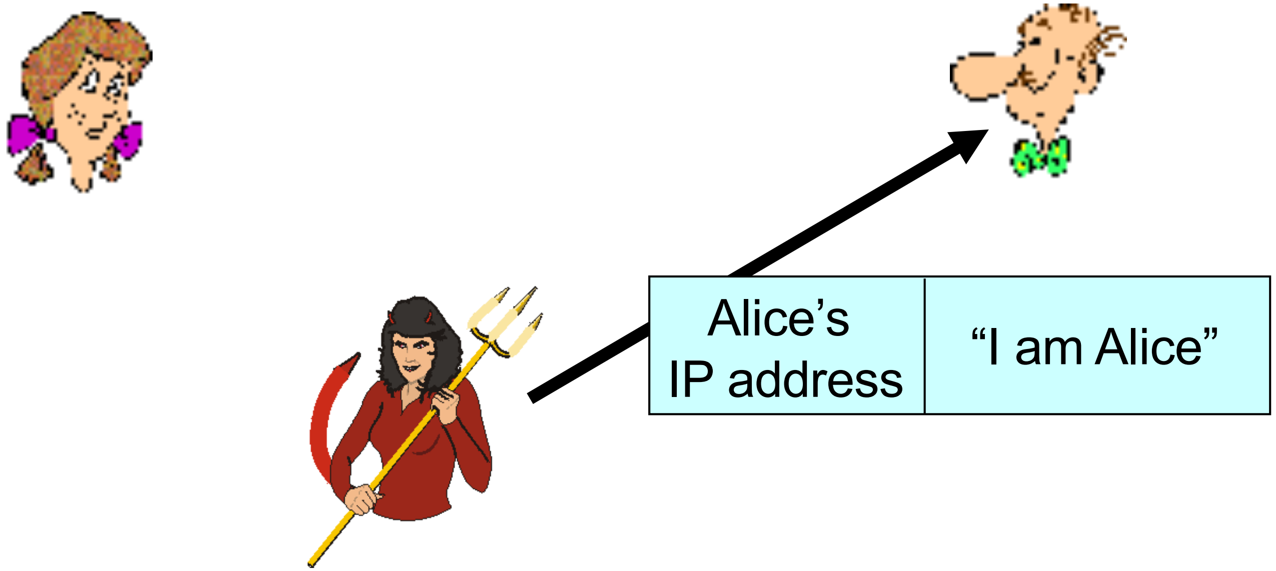
Authentication: another try

Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address

Failure scenario??

Authentication: another try

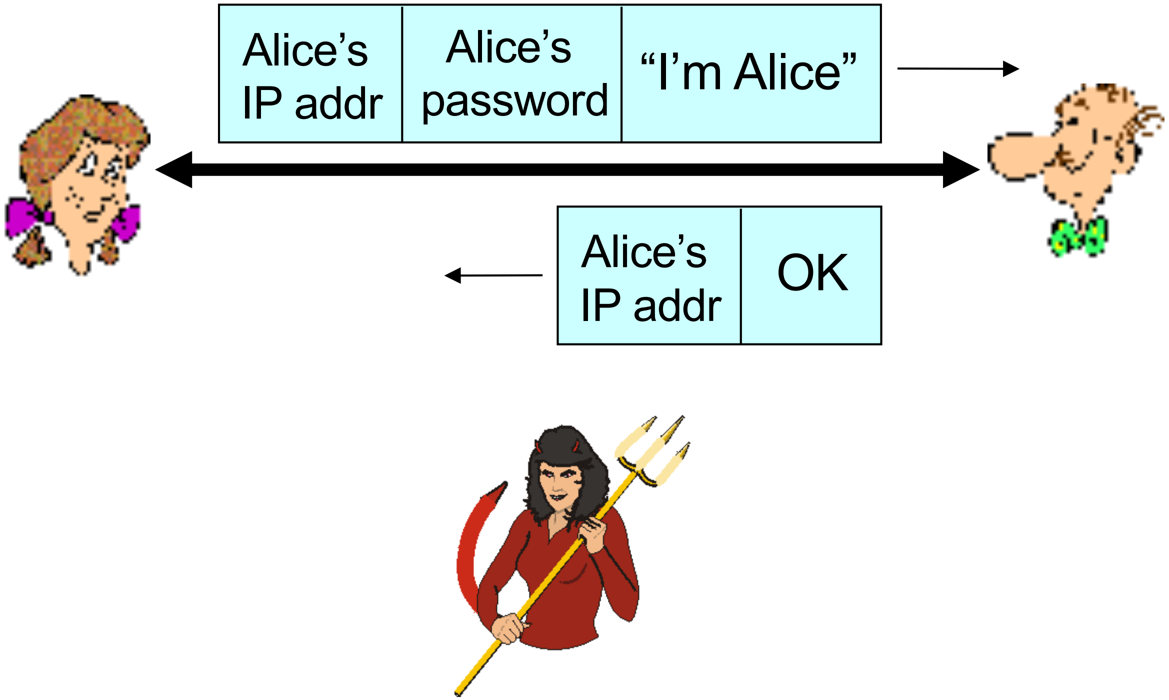
Protocol ap2.0: Alice says “I am Alice” in an IP packet containing her source IP address



Eve can create a packet “spoofing” Alice’s address

Authentication: another try

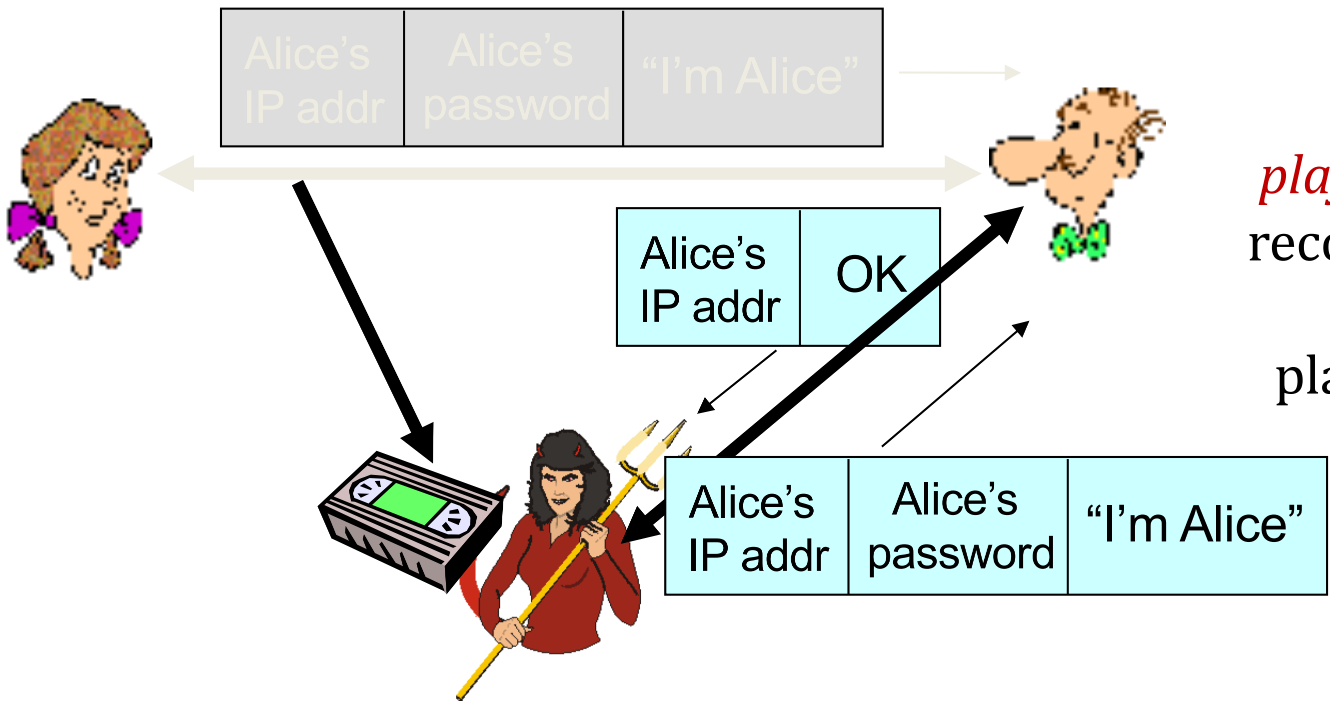
Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



Failure scenario??

Authentication: another try

Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.

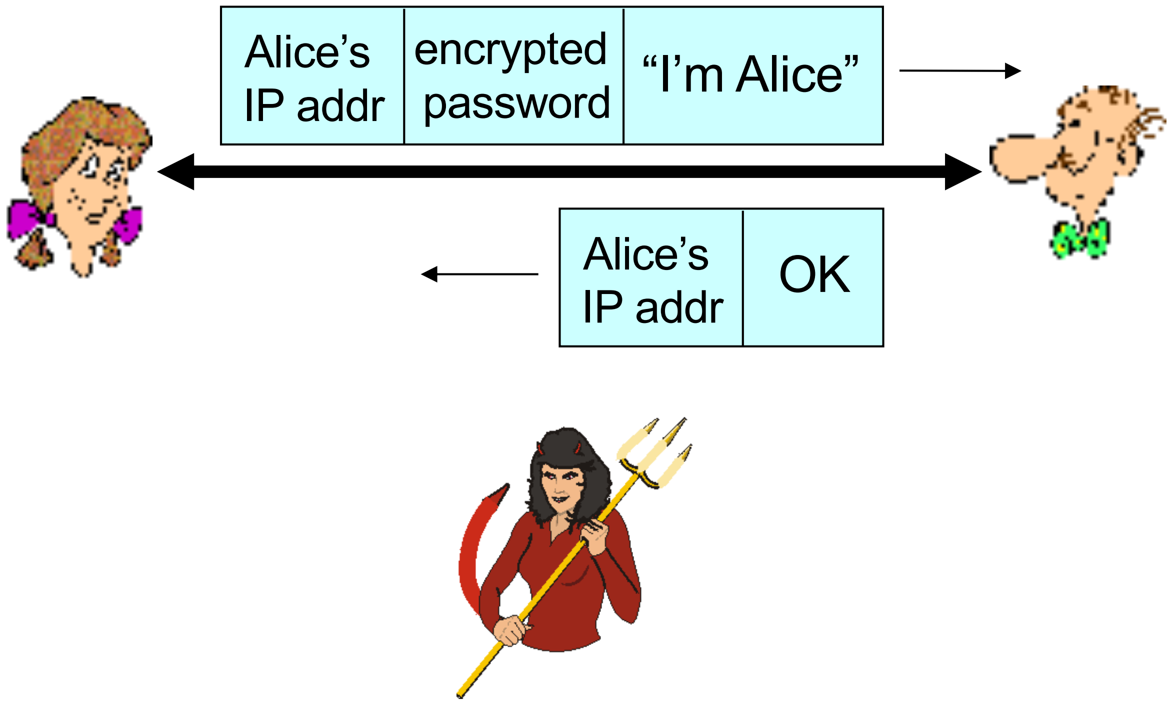


playback attack: Eve records Alice's packet and later plays it back to Bob

UNSW AUSTRALIA
CySPri Laboratory

Authentication: yet another try

Protocol ap3.1: Alice says "I am Alice" and sends her *encrypted* secret password to "prove" it.

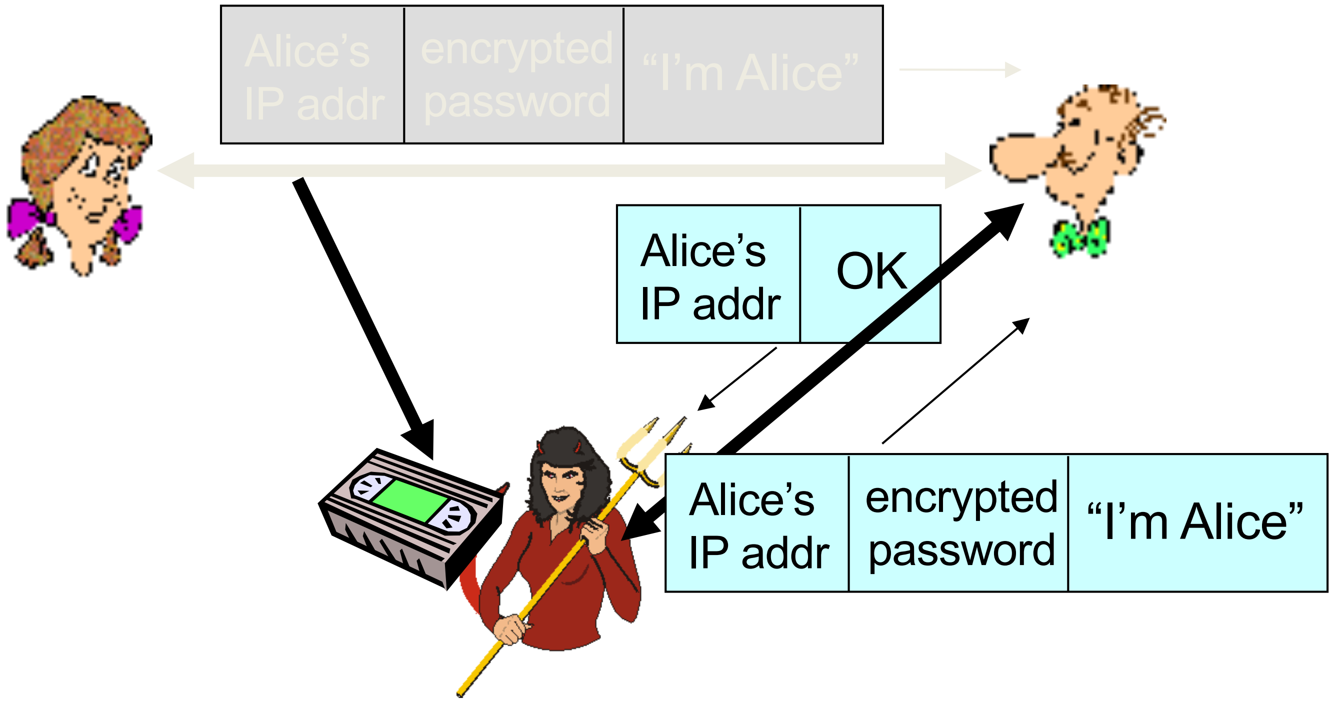


Failure scenario??

UNSW AUSTRALIA
CySPri Laboratory

Authentication: yet another try

Protocol ap3.1: Alice says "I am Alice" and sends her *encrypted* secret password to "prove" it.



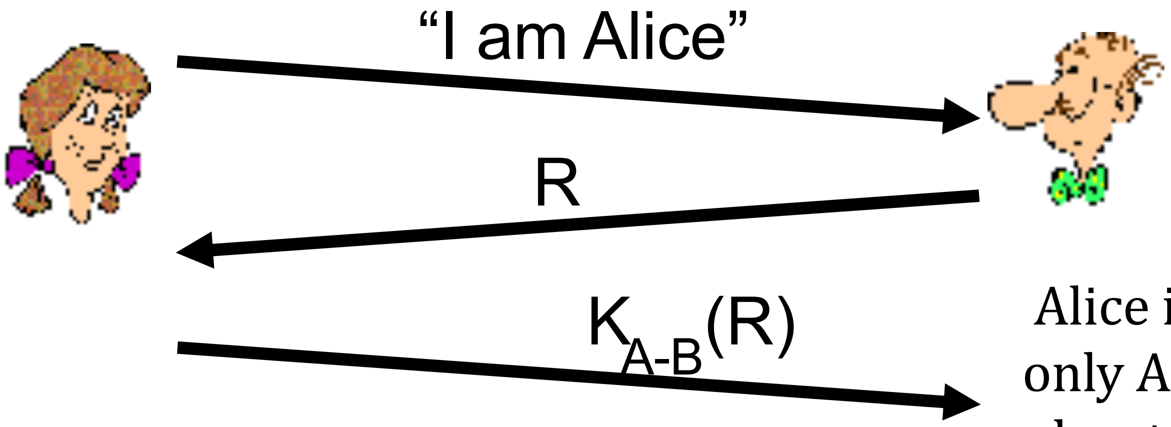
record
and
playback
still works!

Authentication: yet another try

Goal: avoid playback attack

nonce: number (R) used only *once-in-a-lifetime*

ap4.0: to prove Alice "live", Bob sends Alice *nonce*, R. Alice must return R, encrypted with shared secret key



Alice is live, and only Alice knows key to encrypt nonce, so it must be Alice!

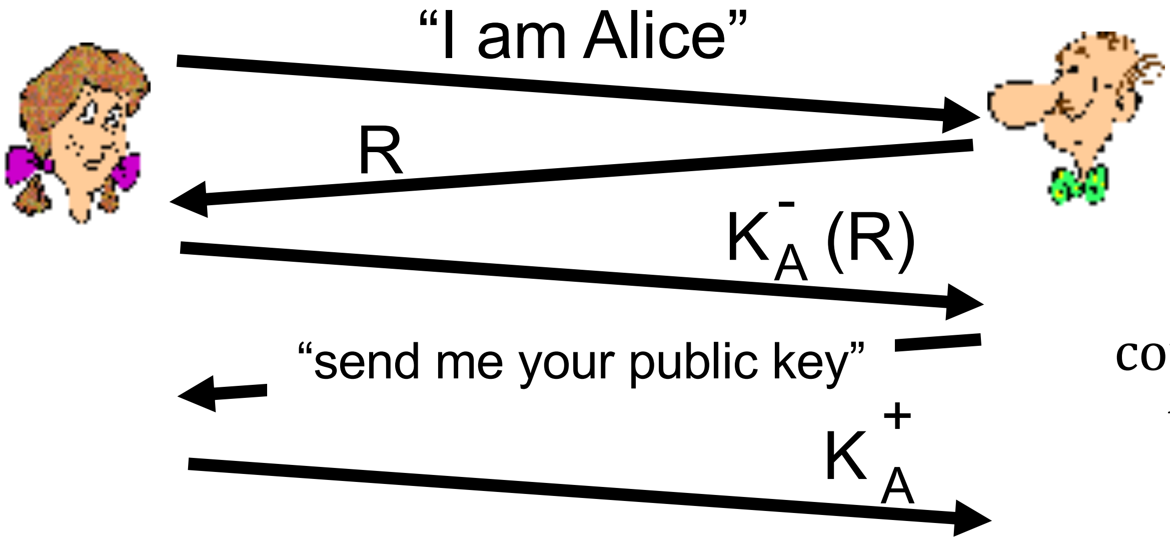
Failures, drawbacks?

Authentication: ap5.0

ap4.0 requires shared symmetric key

- can we authenticate using public key techniques?

ap5.0: use nonce, public key cryptography



Bob computes

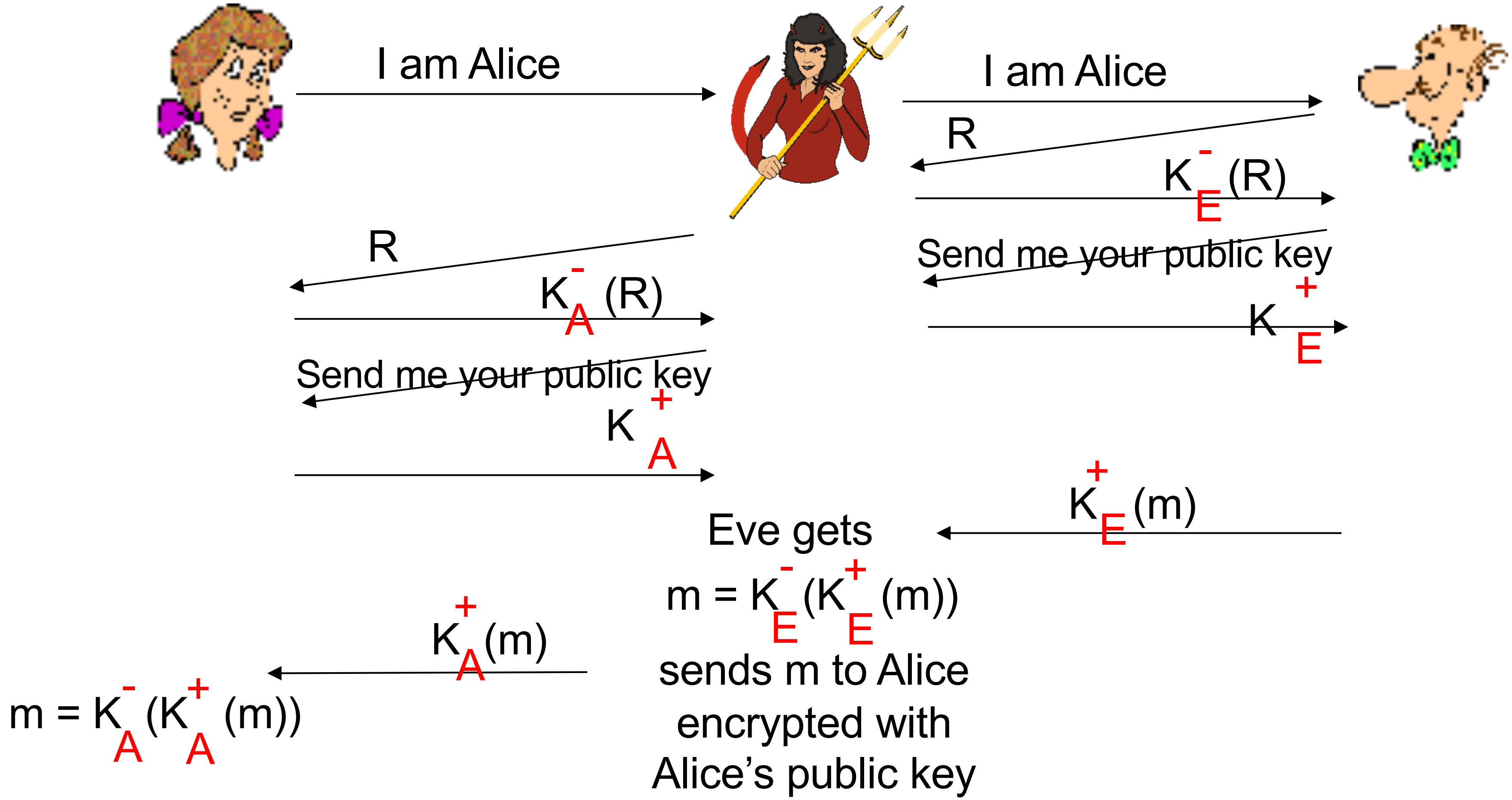
$K_A^+(K_A^-(R)) = R$

and knows only Alice could have the private key, that encrypted R such that

$K_A^+(K_A^-(R)) = R$

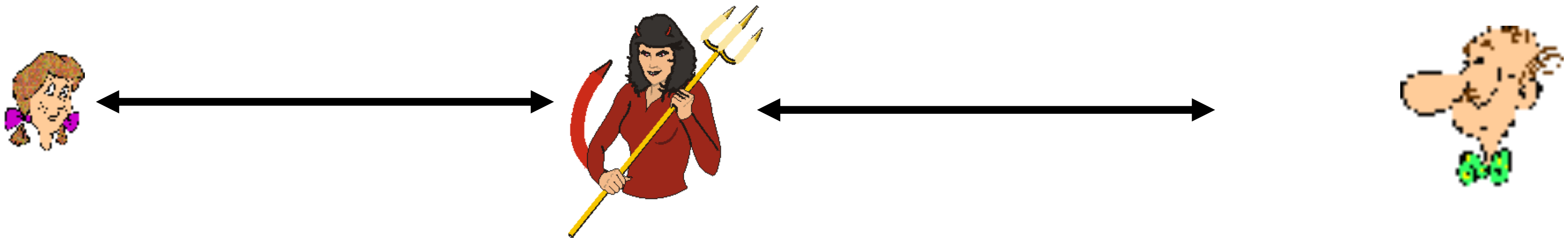
ap5.0: security hole

man (or woman) in the middle attack: Eve poses as Alice (to Bob) and as Bob (to Alice)



ap5.0: security hole

man (or woman) in the middle attack: Eve poses as Alice (to Bob) and as Bob (to Alice)



difficult to detect:

- Bob receives everything that Alice sends, and vice versa. (e.g., so Bob, Alice can meet one week later and recall conversation!)
- problem is that Eve receives all messages as well!

Public key encryption algorithms

Requirements:

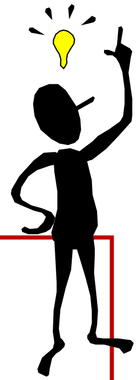
1. need $K_B^{+}(\ )$ and $K_B^{-}(\ )$ such that

$$K_B^{-}(K_B^{+}(m)) = m$$

2. given public key K_B^{+}, it should be impossible to compute private key K_B^{-}

RSA: Rivest, Shamir, Adelson algorithm

Public Key Cryptography



symmetric key crypto

- requires sender, receiver know shared secret key
- Q: how to agree on key in first place (particularly if never "met")?

public key crypto

- radically different approach [Diffie-Hellman76, RSA78]
- sender, receiver do *not* share *secret key*
- *public* encryption key known to *all*
- *private* decryption key known only to receiver

RSA: getting ready

- A message is a bit pattern.
- A bit pattern can be uniquely represented by an integer number.
- Thus encrypting a message is equivalent to encrypting a number.

Example

- m= 10010001. This message is uniquely represented by the decimal number 145.
- To encrypt m, we encrypt the corresponding number, which gives a new number (the ciphertext).

RSA: Creating public/private key pair

1. Choose two large prime numbers *p, q.* (e.g., 1024 bits each)

2. Compute $n = pq$, $z = (p-1)(q-1)$

3. Choose *e* (with $e<n$) that has no common factors with z. (*e, z* are "relatively prime"). E.g.: 4 and 9 are relatively prime. 6 and 9 are not.

4. Choose *d* such that $ed-1$ is exactly divisible by *z*. (in other words: $ed \bmod z = 1$).

5. *Public* key is *(n,e)*. *Private* key is *(n,d)*.

K_B^+ K_B^-

RSA: Encryption, decryption

0. Given (n,e) and (n,d) as computed above

1. To encrypt bit pattern, m $(m < n)$, compute

   $c = m^e \bmod n$ (i.e., remainder when m^e is divided by n)

2. To decrypt received bit pattern, c, compute

   $m = c^d \bmod n$ (i.e., remainder when c^d is divided by n)

Magic happens!

$$m = (\underbrace{m^e \bmod n}_{c})^d \bmod n$$

RSA example

Bob chooses $p=5$, $q=7$. Then $n=35$, $z=24$.

$e=5$ (so e, z relatively prime).

$d=29$ (so $ed-1$ exactly divisible by z).

encrypt:

letter	m	m^e	$c = m^e \bmod n$
l	12	1524832	17

decrypt:

c	c^d	$m = c^d \bmod n$	letter
17	481968572106750915091411825223071697	12	l

RSA: another important property

The following property will be *very* useful later:

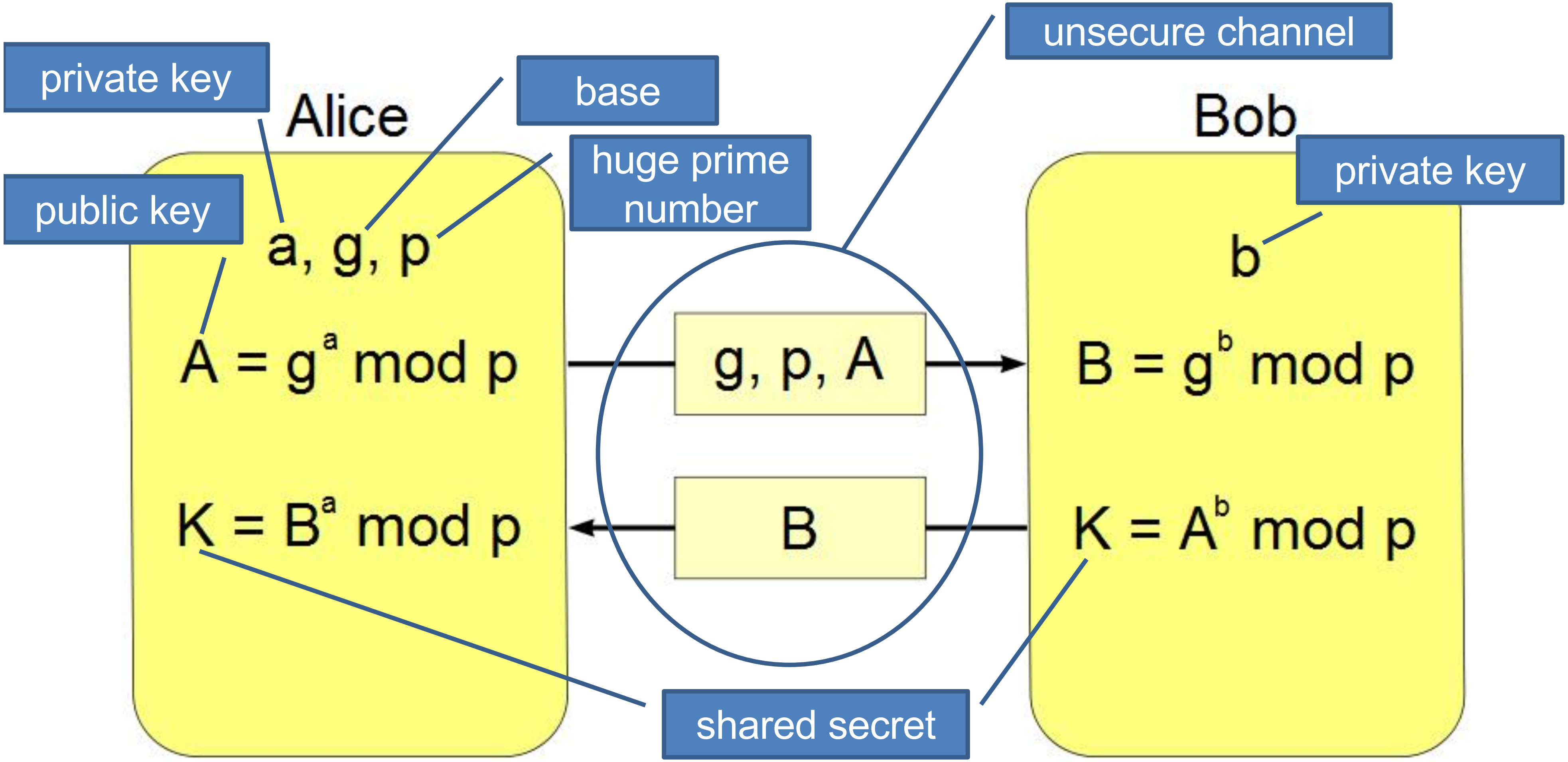
$$K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$$

use public key first, followed by private key

use private key first, followed by public key

Result is the same!

Diffie-Hellman key exchange



$$K = A^b \bmod p = (g^a \bmod p)^b \bmod p = g^{ab} \bmod p = (g^b \bmod p)^a \bmod p = B^a \bmod p$$

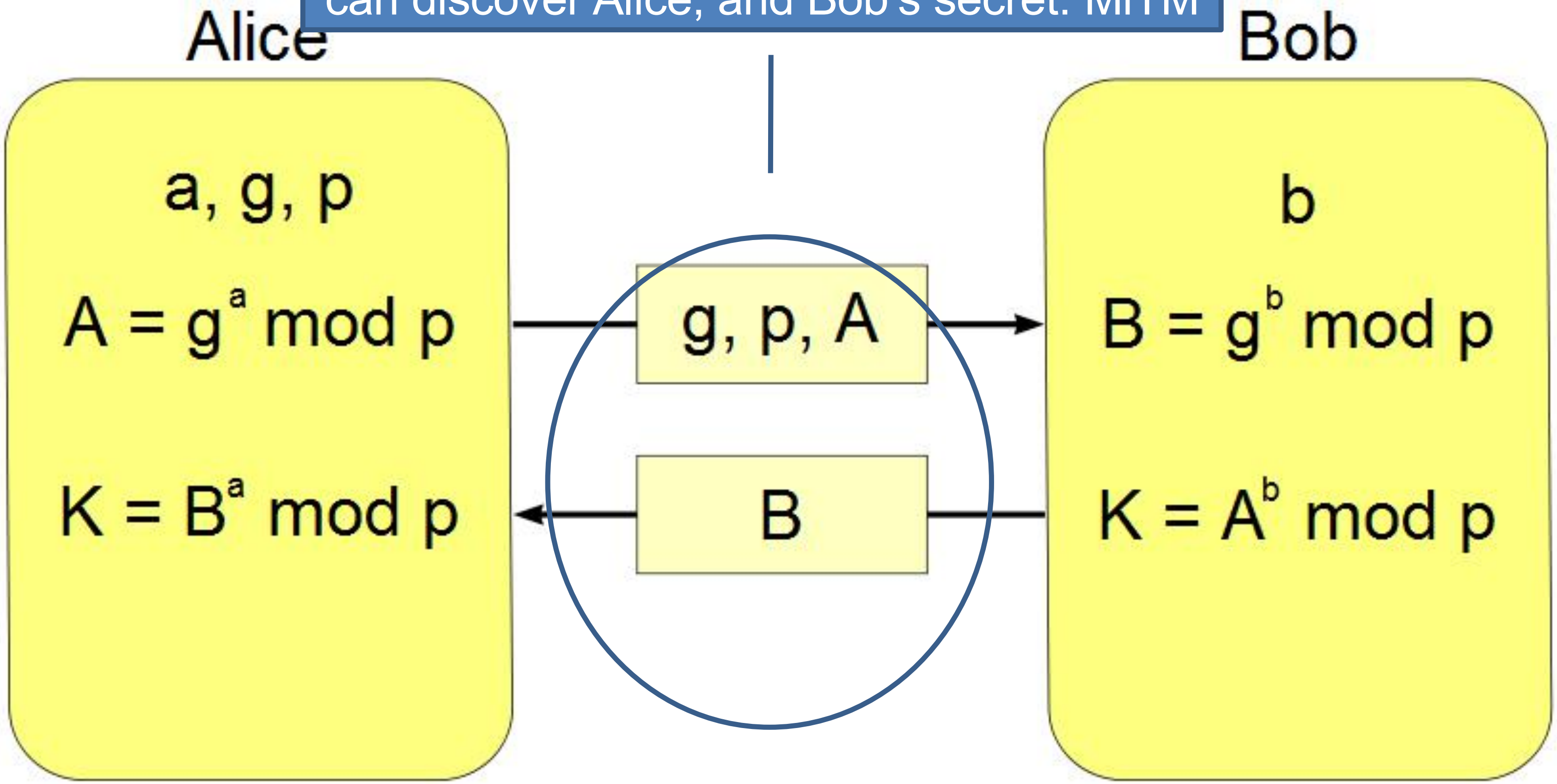
Alice’s private key = 5, Bob’s private key = 4, g=3, p=7

Alice’s public key = 3^5 mod 7 = 5, Bob’s public key = 3^4 mod 7 = 4

Alice’s shared key = 4^5 mod 7 = 2, Bob’s shared key = 5^4 mod 7 = 2

Diffie-Hellman key exchange

If Eve can tamper with the channel, she can discover Alice, and Bob's secret: MiTM



$K = A^b \bmod p = (g^a \bmod p)^b \bmod p = g^{ab} \bmod p = (g^b \bmod p)^a \bmod p = B^a \bmod p$

Alice's private key = 5, Bob's private key = 4, g=3, p=7

Alice's public key = 3^5 mod 7 = 5, Bob's public key = 3^4 mod 7 = 4

Alice's shared key = 4^5 mod 7 = 2, Bob's shared key = 5^4 mod 7 = 2

Formal Analysis of Security Protocols

- Engineers/developers take a reactive approach
  - Design protocols for known attack vectors
  - Fix problems after attacks have actually happened or Zero day
- Formal Analysis can aid improving security
  - Not full proof but can discover many security holes
  - Take remedial action during design/implementation phase
  - Example: Formal analysis showed vulnerabilities in SSL/TLS record protocols
- Tools and techniques vary in their strength and sophistication, some are simpler to use, others have more advanced features with steep learning curve
  - Complex tools: Tamarin, Scyther, Proverif ….. List goes on
  - Simpler tool: AVISPA, lot of examples of Internet protocols
- We will introduce AVISPA for appreciation, if this interests you, you can explore others in your own time
- You can find more here http://www.avispa-project.org/

Acknowledgements

- Network Security Essentials: Stallings, Chapter 4 provided by Henric Johnson, Blekinge Institute of Technology,Sweden (Please refer to Section 4.3 and 4.4 from Staillings)
- Computer Networking A top-Down Approach: Jim Kurose and Keith Ross, chapter 8 (several lecture foils provided by authors)
- http://www.avispa-project.org/
- A nice youtube tutorial that we use in lecture
  - https://www.youtube.com/watch?v=YvgHw5pr5bA