2

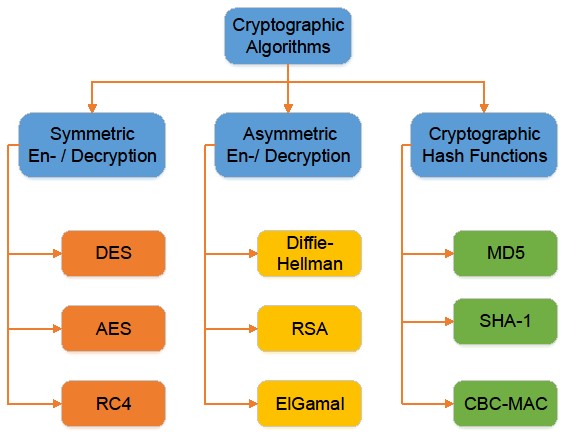
# Today’s Agenda

* Authentication Recap
* Key distribution using asymmetric encryption

– Public-key distribution of secret keys

* Formal Method for Protocol Specification and Verification: AVISPA Tool

# Recap



4

# Recap Authentication Basics

* Quick recap, possibly already done in

Computer Networks

* 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??



“

I am Alice

”



# Authentication

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

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

In a network,



“

I am Alice

”

Bob can not “see” Alice, so

Eve simply declares herself to be Alice

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

Failure scenario??



“

I am Alice

”

Alice

’

s

IP address

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

Eve can create a packet “spoofing”



“

I am Alice

”

Alice

’

s

IP address

Alice’s address

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

Failure scenario??



“

I

’

m Alice

”

Alice

’

s

IP addr

Alice

’

s

password

OK

Alice

’

s

IP addr

*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



“

I

’

m Alice

”

Alice

’

s

IP addr

Alice

’

s

password

OK

Alice

’

s

IP addr

“

I

’

m Alice

”

Alice

’

s

IP addr

Alice

’

s

password

*Diagram

Description automatically generatedProtocol ap3.1:* Alice says “I am Alice” and sends her *encrypted* secret password to “prove” it.

record

and

playback

*still*

works!



“

I

’

m Alice

”

Alice

’

s

IP addr

encrypted

password

OK

Alice

’

s

IP addr

“

I

’

m Alice

”

Alice

’

s

IP addr

encrypted

password

# Diagram Description automatically generated

# Diagram Description automatically generatedDiagram Description automatically generatedDiagram Description automatically generated

# 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:

v Bob receives everything that Alice sends, and vice versa. (e.g., so

Bob, Alice can meet one week later and recall conversation!) v problem is that Eve receives all messages as well!

# Text Description automatically generatedPublic 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* v radically different approach [Diffie-

Hellman76, RSA78] v 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)

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

1. Choose *d* such that *ed-1* is exactly divisible by *z*. (in other words: *ed* mod *z = 1* ).
2. *Public* key is *(n,e). Private* key is *(n,d).*

KB+ KB-

# RSA: Encryption, decryption

1. Given (*n,e*) and (*n,d*) as computed above
2. To encrypt bit pattern, *m (m < n)*, compute *c = m* emod *n* (i.e., remainder when *m*eis divided by *n*)
3. To decrypt received bit pattern, *c*, compute *m = c* dmod *n* (i.e., remainder when *c*dis divided by *n*)

|  |
| --- |
| Magic m = (m emod n)dmod n happens! c |

# 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).

letter m me c = m mod ne

encrypt:

l 12 1524832 17

decrypt: c cd m = c mod nd letter

17 481968572106750915091411825223071697 12 l

# RSA: another important property

The following property will be *very* useful later:

K (K (m)) = m = K

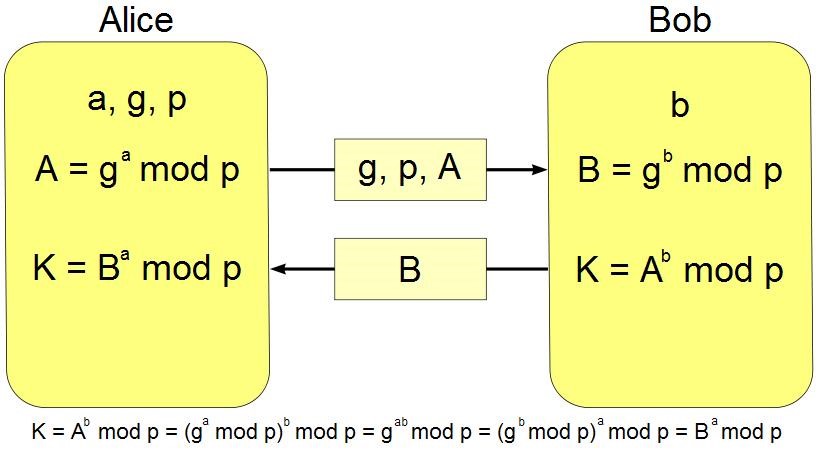
B- B+ +B(K (m)-B )

use public key use private key

first, followed by first, followed by private key public key

*Result is the same!*

# Diffie-Hellman key exchange



base

huge prime

number

private key

public key

unsecure channel

private key

shared secret

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

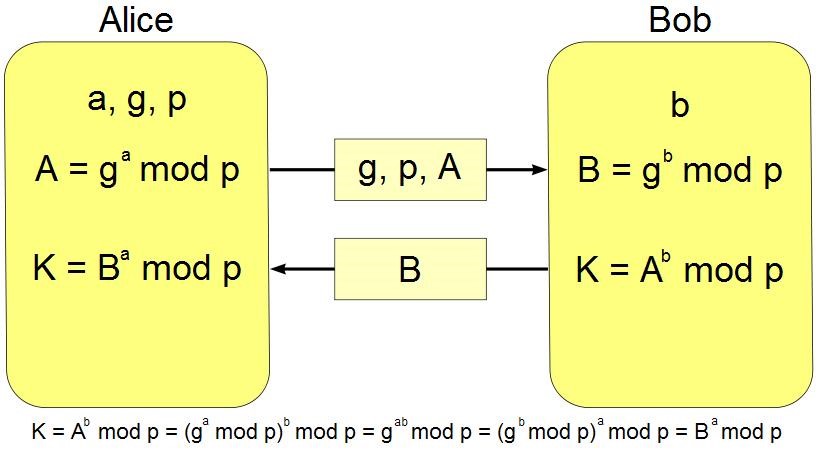
mod 7 = 2, Bob’s shared key = 5

4

mod 7 = 2

# Diffie-Hellman key exchange

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



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

26

# 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/

•