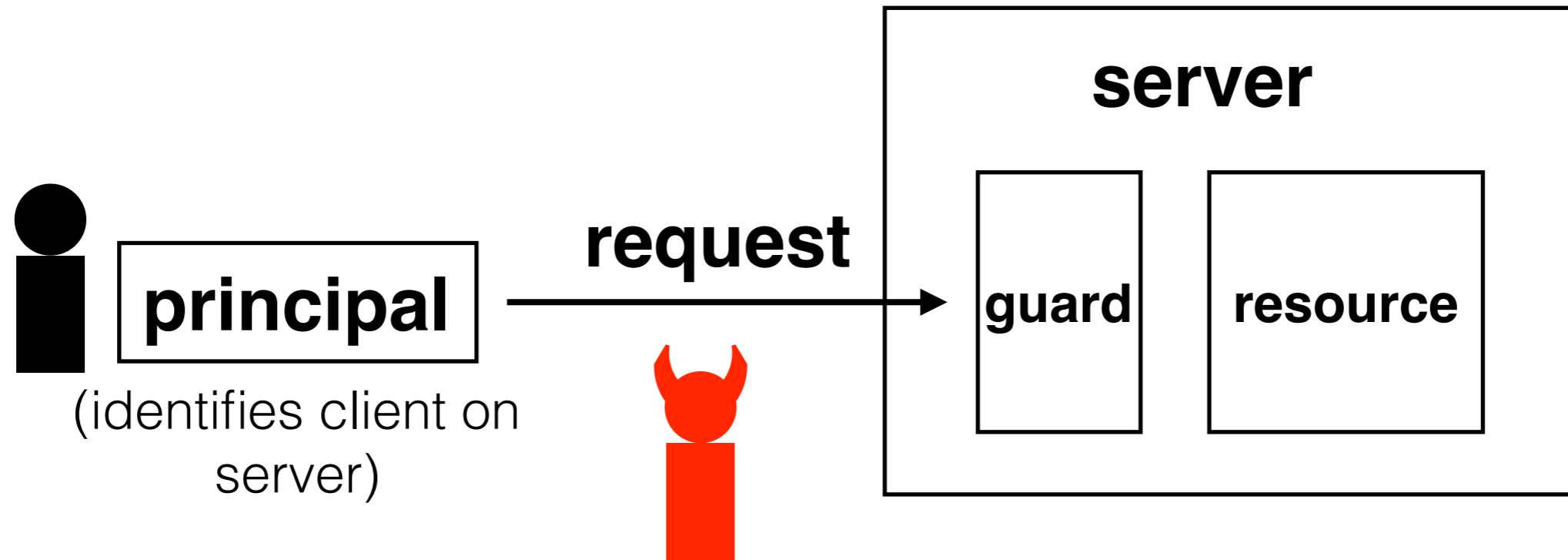


# 6.033 Spring 2018

## Lecture #22

- **Combating network adversaries**
  - **Secure Channels**
  - **Signatures**



```

14:49:19.858386 2805536312us tsft -95dB noise antenna 1 5785 MHz 11a ht/40+ [bit 20] CF +QoS IP
17.253.11.201.80 > 10.189.53.19.54191: Flags [.], seq 3088997:3090365, ack 0, win 124, options [nop,nop,TS
val 295799082 ecr 1238603892], length 1368: HTTP
 0x0000:  aaaa 0300 0000 0800 4500 058c 37fd 4000  .....E...7.@.
 0x0010:  3b06 a4d9 11fd 0bc9 0abd 3513 0050 d3af  ;.....5..P..
 0x0020:  f692 6b9d 0186 6995 8010 007c 60b6 0000  ..k...i....|`...
 0x0030:  0101 080a 11a1 892a 49d3 9874 626a 6563  .....*I..tjtec
 0x0040:  7473 2e6e 6962 2e6d 6574 6155 5808 00e3  ts.nib.metaUX...
 0x0050:  8ee3 5a89 29e3 5a50 4b01 021e 0314 0000  ..Z.)ZPK.....
 0x0060:  0863 00b7 359b 4c5e bd8f e3c1 0900 00e9  .c..5.L^.....
 0x0070:  1200 0079 000c 0000 0000 0000 0000 40a4  ...y.....@.
 0x0080:  814c ab1c 0650 6179 6c6f 6164 2f68 696c  .L...Payload/hil
 0x0090:  6c64 6173 6832 2e61 7070 2f48 7355 4952  ldash2.app/HsUIR
 0x00a0:  6573 6f75 7263 6542 756e 646c 652e 6275  essageBundle.bu
 0x00b0:  6e64 6c65 2f68 7353 7570 706f 7274 4d61  ndle/hsSupportMa
 0x00c0:  696e 2e73 746f 7279 626f 6172 6463 2f78  in.storyboardc/x
 0x00d0:  7965 2d32 722d 456a 6b2d 7669 6577 2d38  ye-2r-Ejk-view-8
 0x00e0:  394e 2d70 532d 3437 647e 6970 6164 2e6e  9N-pS-47d~ipad.n
 0x00f0:  6962 2f72 756e 7469 6d65 2e6e 6962 5558  ib/runtime.nibUX
 0x0100:  0800 e38e e35a 8929 e35a 504b 0102 1e03  ....Z.)ZPK....
 0x0110:  1400 0008 0000 b735 9b4c 5cf6 7335 8500  .....5.L\.s5..
 0x0120:  0000 8500 0000 7e00 0c00 0000 0000 0000  .....~.....
 0x0130:  0040 a481 b4b5 1c06 5061 796c 6f61 642f  .@.....Payload/
 0x0140:  6869 6c6c 6461 7368 322e 6170 702f 4873  hilldash2.app/Hs
 0x0150:  5549 5265 736f 7572 6365 4275 6e64 6c65  UIResourceBundle
 0x0160:  2e62 756e 646c 652f 6873 5375 7070 6f72  .bundle/hsSuppor
 0x0170:  744d 6169 6e2e 7374 6f72 7962 6f61 7264  tMain.storyboard
 0x0180:  632f 7879 652d 3272 2d45 6a6b 2d76 6965  c/xye-2r-Ejk-vie
 0x0190:  772d 3839 4e2d 7053 2d34 3764 7e69 7061  w-89N-pS-47d~ipa

```

14:15:57.156383 731851825us tsft -95dB noise antenna 0 2412 MHz 11g ht/20 26.0 Mb/s MCS 3 20 MHz lon GI  
greenfield BCC FEC [bit 20] CF +QoS IP dhcp-18-111-89-99  
.dyn.mit.edu.57061 > 17.154.66.156.https: Flags [P.], seq 0:517, ack 1, win 8192, length 517

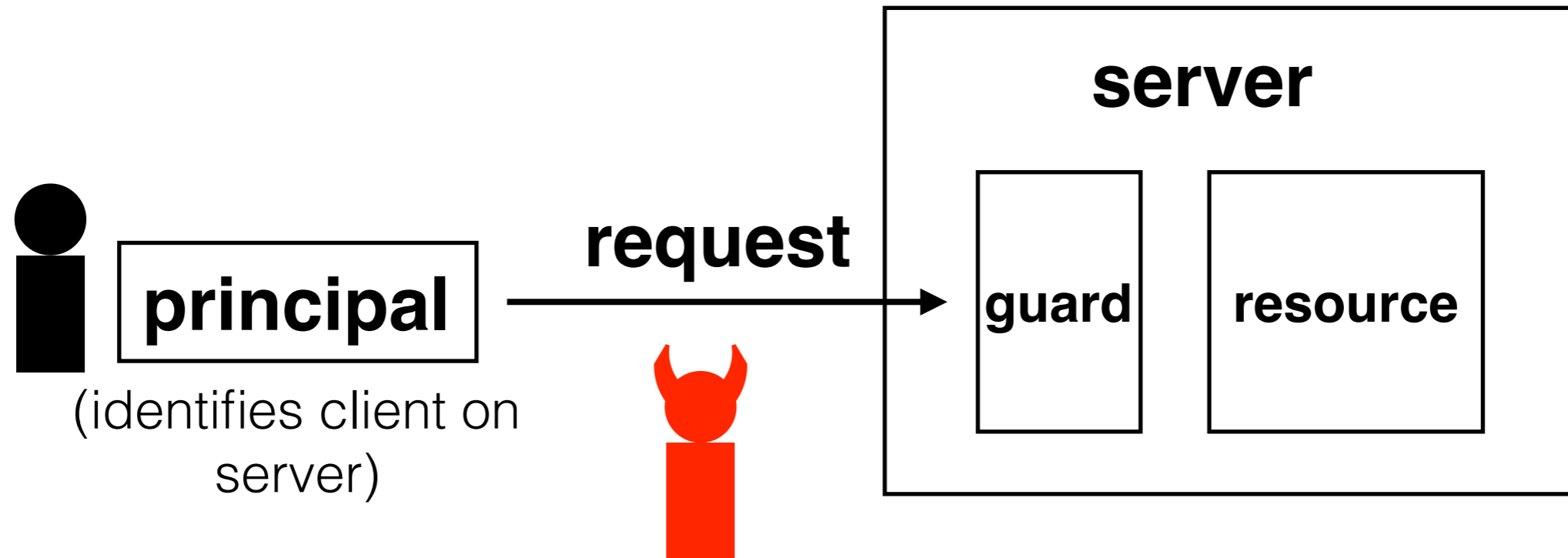
```
0x0000:  aaaa 0300 0000 0800 4500 022d 9fd8 4000  .....E...-..@.  
0x0010:  4006 d8ea 126f 5963 119a 429c dee5 01bb  @....oYc..B.....  
0x0020:  f7f4 9d92 e59a 1614 5018 2000 ae38 0000  .....P....8..  
0x0030:  1603 0102 0001 0001 fc03 0359 077b 5d64  .....Y.{}]d  
0x0040:  6a53 0208 0cde 5c0a 26e8 5732 151d c778  jS....\.&.W2...x  
0x0050:  16c3 d1cc d5e6 c8a1 b940 3220 3ce6 c3c9  .....@2.<...  
0x0060:  ccb5 f523 3ae1 bf92 cd1f 1ac9 efc4 b155  ...#:.U  
0x0070:  576a 4af8 4bc9 5b38 38dd 5d0e 0026 00ff  WjJ.K.[88.]..&..  
0x0080:  c02c c02b c024 c023 c00a c009 c030 c02f  .,+.$.#....0./  
0x0090:  c028 c027 c014 c013 009d 009c 003d 003c  .('.....=<  
0x00a0:  0035 002f 0100 018d 0000 001d 001b 0000  .5./.....  
0x00b0:  1870 3331 2d62 7579 2e69 7475 6e65 732e  .p31-buy.itunes.  
0x00c0:  6170 706c 652e 636f 6d00 0a00 0800 0600  apple.com.....  
0x00d0:  1700 1800 1900 0b00 0201 0000 0d00 1200  .....  
0x00e0:  1004 0102 0105 0106 0104 0302 0305 0306  .....  
0x00f0:  0333 7400 0000 1000 3000 2e02 6832 0568  .3t....0...h2.h  
0x0100:  322d 3136 0568 322d 3135 0568 322d 3134  2-16.h2-15.h2-14  
0x0110:  0873 7064 792f 332e 3106 7370 6479 2f33  .spdy/3.1.spdy/3  
0x0120:  0868 7474 702f 312e 3100 0500 0501 0000  .http/1.1.....  
0x0130:  0000 0012 0000 0017 0000 0015 00f7 0000  .....  
0x0140:  0000 0000 0000 0000 0000 0000 0000 0000  .....  
0x0150:  0000 0000 0000 0000 0000 0000 0000 0000  .....  
0x0160:  0000 0000 0000 0000 0000 0000 0000 0000  .....  
0x0170:  0000 0000 0000 0000 0000 0000 0000 0000  .....  
0x0180:  0000 0000 0000 0000 0000 0000 0000 0000  .....  
0x0190:  0000 0000 0000 0000 0000 0000 0000 0000  .....  
0x01a0:  0000 0000 0000 0000 0000 0000 0000 0000  .....  
0x01b0:  0000 0000 0000 0000 0000 0000 0000 0000  .....  
0x01c0:  0000 0000 0000 0000 0000 0000 0000 0000  .....  
0x01d0:  0000 0000 0000 0000 0000 0000 0000 0000  .....  
0x01e0:  0000 0000 0000 0000 0000 0000 0000 0000  .....  
0x01f0:  0000 0000 0000 0000 0000 0000 0000 0000  .....
```

14:05:50.087089 195784191us tsft bad-fcs -78dB signal -96dB noise antenna 1 5785 MHz 11a ht/40+ [bit 20] CF +QoS IP 18.111.23.61.64677 > 104.199.110.216.80: Flag s [P.], seq 1:323, ack 1, win 4136, options [nop,nop,TS val 605691701 ecr 1821306901], length 322: HTTP: GET /img/inj9/b/p0k/x6jl.png HTTP/1.1

0x0000:	aaaa 0300 0000 0800 4500 0176 a863 4000	.....E..v.c@.
0x0010:	4006 8fd3 126f 173d 68c7 6ed8 fca5 0050	@....o.=h.n....P
0x0020:	9d4a 295a 0fc9 838f 8018 1028 b54f 0000	.J)Z.....(.0..
0x0030:	0101 080a 241a 1f35 6c8e f015 4745 5420	....\$.5l...GET.
0x0040:	2f69 6d67 2f69 6e6a 392f 622f 7030 6b2f	/img/inj9/b/p0k/
0x0050:	7836 6a6c 2e70 6e67 2048 5454 502f 312e	x6jl.png.HTTP/1.
0x0060:	310d 0a48 6f73 743a 2069 6e6a 392e 6d6a	1..Host:.inj9.mj
0x0070:	742e 6c75 0d0a 4163 6365 7074 3a20 696d	t.lu..Accept:.im
0x0080:	6167 652f 706e 672c 696d 6167 652f 7376	age/png,image/sv
0x0090:	672b 786d 6c2c 696d 6167 652f 2a3b 713d	g+xml,image/*;q=
0x00a0:	302e 382c 2a2f 2a3b 713d 302e 350d 0a41	0.8,*/*;q=0.5..A
0x00b0:	6363 6570 742d 4ce1 4d67 7561 6765 3a20	ccept-L.Mguage:.
0x00c0:	656e 2d75 730d 0a43 6f6e 6e65 6374 696f	en-us..Connectio
0x00d0:	6e3a 206b 6565 702d 616c 6976 650d 0a41	n:.keep-alive..A
0x00e0:	6363 6570 742d 456e 636f 6469 6e67 3a20	ccept-Encoding:.
0x00f0:	677a 6970 a18c 7b65 666c 6174 650d 0a55	gzip..{eflate..U
0x0100:	7365 722d 4167 656e 743a 204d 6f7a 696c	ser-Agent:.Mozil
0x0110:	6c61 2f35 2e30 2028 6950 686f 6e65 3b20	la/5.0.(iPhone;.
0x0120:	4350 5520 6950 686f 6e65 204f 5320 3130	CPU.iPhone.OS.10
0x0130:	5f33 5f31 206c 696b 6520 4d61 6320 4f53	_3_1.like.Mac.OS
0x0140:	2058 2920 4170 706c 6557 6562 4b69 742f	.X).AppleWebKit/
0x0150:	3630 332e 312e 3330 2028 4b48 544d 4c2c	603.1.30.(KHTML,
0x0160:	206c 696b 6520 4765 636b 6f29 204d 6f62	.like.Gecko).Mob
0x0170:	696c 652f 3134 4533 3034 0d0a 0d0a	ile/14E304....

14:05:29.947459 104653458us tsft -70dB signal -92dB noise antenna 0 2412 MHz 11g ht/20 39.0 Mb/s MCS 10 20 MHz lon GI mixed BCC FEC [bit 20] CF +QoS IP 10.189.6.135.5353 > 224.0.0.251.5353: 0\*- [0q] 2/0/3 (Cache flush) PTR Bobs-iPhone.local., (Cache flush) PTR Bobs-iPhone.local. (217)

```
0x0000:  aaaa 0300 0000 0800 4500 00f5 2053 0000  .....E....S..
0x0010:  ff11 a865 0abd 0687 e000 00fb 14e9 14e9  ...e.....
0x0020:  00e1 5867 0000 8400 0000 0002 0000 0003  ..Xg.....
0x0030:  0137 0135 0144 0133 0139 0130 0138 0133  .7.5.D.3.9.0.8.3
0x0040:  0135 0135 0139 0144 0144 0141 0143 0130  .5.5.9.D.D.A.C.0
0x0050:  0130 0130 0130 0130 0130 0130 0130 0130  .0.0.0.0.0.0.0
0x0060:  0130 0130 0130 0130 0130 0138 0145 0146  .0.0.0.0.0.8.E.F
0x0070:  0369 7036 0461 7270 6100 000c 8001 0000  .ip6.arpa.....
0x0080:  0078 0015 0d44 3139 8b64 432d 6950 686f  .x.....Bobs-iPho
0x0090:  6e65 056c 6f63 616c 0003 3133 3501 3603  ne.local..135.6.
0x00a0:  3138 3902 3130 0769 6e2d 6164 6472 c050  189.10.in-addr.P
0x00b0:  000c 8001 0000 0078 0002 c060 c00c 002f  .....x...`.../
0x00c0:  8001 0000 0078 0006 c00c 0002 0008 c075  .....x.....u
0x00d0:  002f 8001 0000 0078 0006 c075 0002 0008  ./.....x...u....
0x00e0:  0000 2905 a000 0011 9400 1200 0400 0e00  ..).....
0x00f0:  256e 8dc1 7d01 b16c 8dc1 7d01 b1      %n..}.1..}..
```



**confidentiality:** adversary cannot learn message contents

**integrity:** adversary cannot tamper with message contents  
(if they do, client and/or server will detect it)

**encrypt**(**key**, **message**) → **ciphertext**  
**decrypt**(**key**, **ciphertext**) → **message**

encrypt(34fbcbd1, "hello, world") = 0x47348f63a67926cd393d4b93c58f78c  
decrypt(34fbcbd1, "0x47348f63a67926cd393d4b93c58f78c") = hello, world

**property:** given the **ciphertext**, it is (virtually) impossible to obtain the **message** without knowing the **key**



adversary can't determine **message**, *but* might be able to cleverly alter **ciphertext** so that it decrypts to a different message



**encrypt**(key, message) → ciphertext  
**decrypt**(key, ciphertext) → message

encrypt(34fbcbd1, "hello, world") = 0x47348f63a67926cd393d4b93c58f78c  
decrypt(34fbcbd1, "0x47348f63a67926cd393d4b93c58f78c") = hello, world

**property:** given the **ciphertext**, it is (virtually) impossible to obtain the **message** without knowing the **key**



no good — if the adversary changes **ciphertext**, it can also (correctly) update the hash

**encrypt**(key, message) → ciphertext  
**decrypt**(key, ciphertext) → message

encrypt(34fbcbd1, "hello, world") = 0x47348f63a67926cd393d4b93c58f78c  
decrypt(34fbcbd1, "0x47348f63a67926cd393d4b93c58f78c") = hello, world

**property:** given the ciphertext, it is (virtually) impossible to obtain the message without knowing the key

**MAC**(key, message) → token

MAC(34fbcbd1, "hello, world") = 0x59cccc95723737f777e62bc756c8da5c

**property:** given the message, it is (virtually) impossible to obtain the token without knowing the key

(it is also impossible to go in the reverse direction)

alice

$c = \text{encrypt}(k, m)$

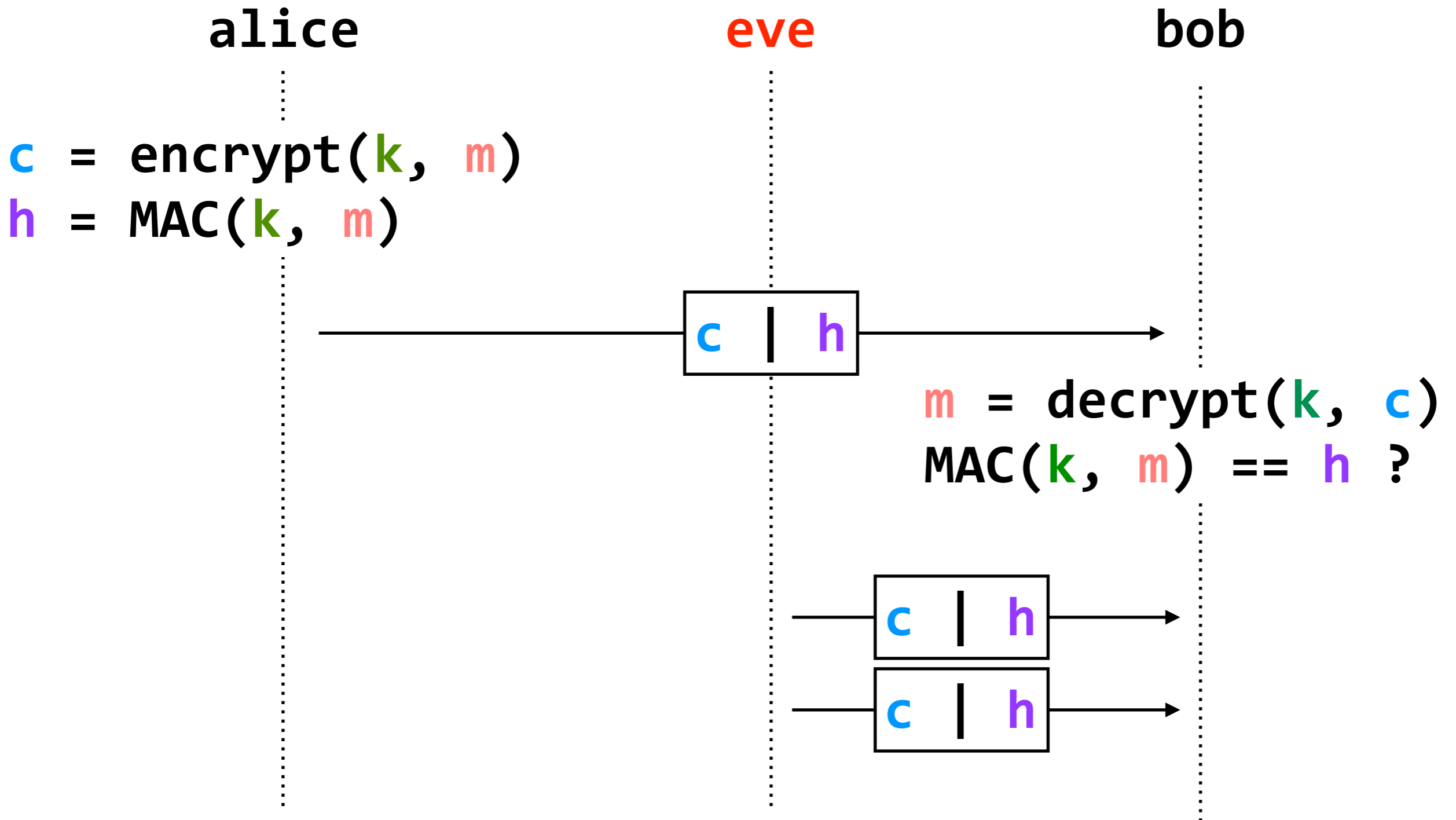
$h = \text{MAC}(k, m)$

bob



$m = \text{decrypt}(k, c)$

$\text{MAC}(k, m) == h ?$



alice

bob

$c = \text{encrypt}(k, m \mid \text{seq})$

$h = \text{MAC}(k, m \mid \text{seq})$



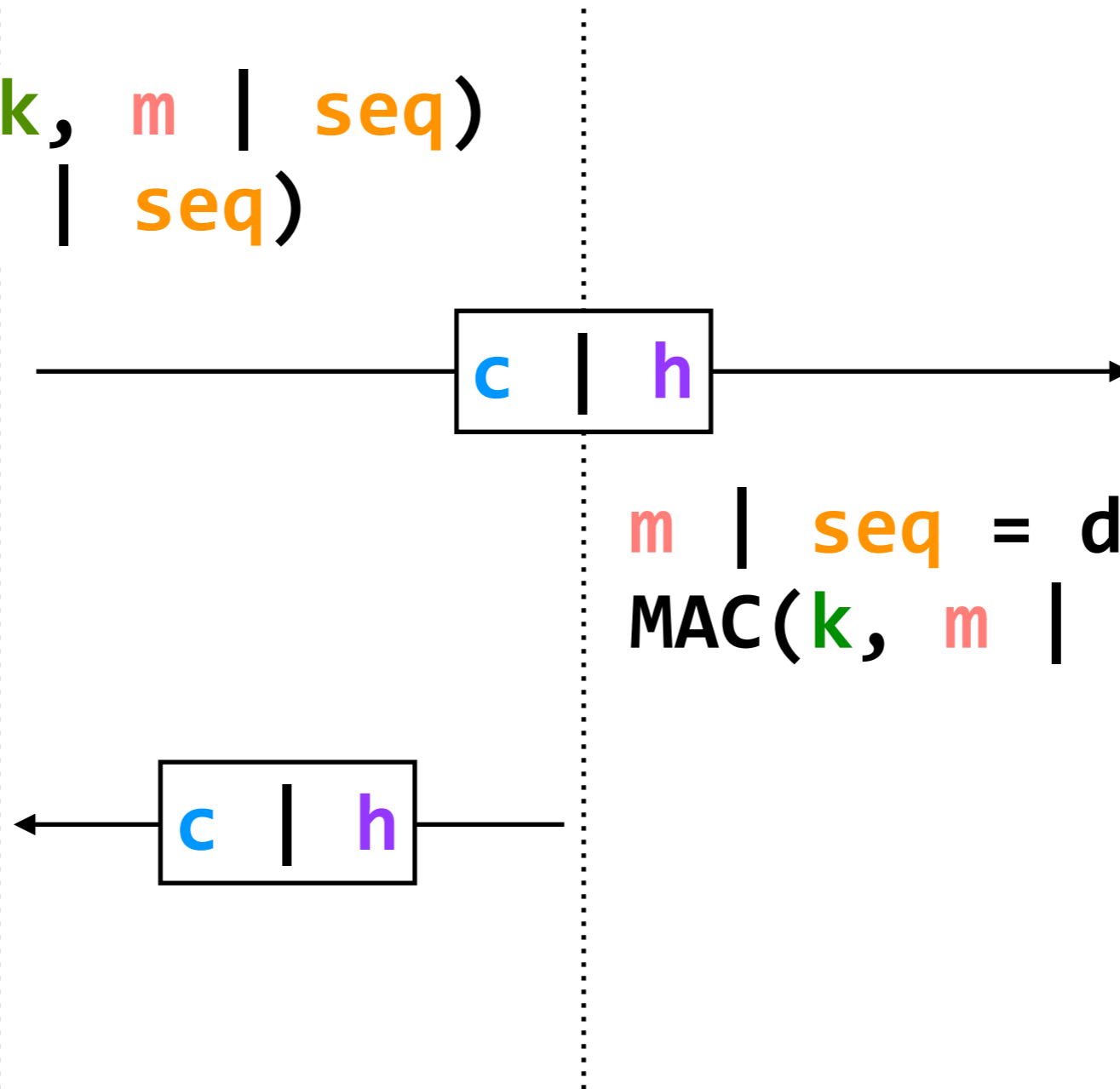
$m \mid \text{seq} = \text{decrypt}(k, c)$   
 $\text{MAC}(k, m \mid \text{seq}) == h ?$

alice

eve

bob

$c = \text{encrypt}(k, m \mid \text{seq})$   
 $h = \text{MAC}(k, m \mid \text{seq})$



$m \mid \text{seq} = \text{decrypt}(k, c)$   
 $\text{MAC}(k, m \mid \text{seq}) == h ?$

**problem:** reflection attacks

(adversary could intercept a message, re-send it at a later time in the opposite direction)

alice

bob

$$c_a = \text{encrypt}(k_a, m_a \mid \text{seq}_a)$$

$$h_a = \text{MAC}(k_a, m_a \mid \text{seq}_a)$$



$$m_a \mid \text{seq}_a = \text{decrypt}(k_a, c_a)$$
$$\text{MAC}(k_a, m_a \mid \text{seq}_a) == h_a ?$$

$$c_b = \text{encrypt}(k_b, m_b \mid \text{seq}_b)$$
$$h_b = \text{MAC}(k_b, m_b \mid \text{seq}_b)$$

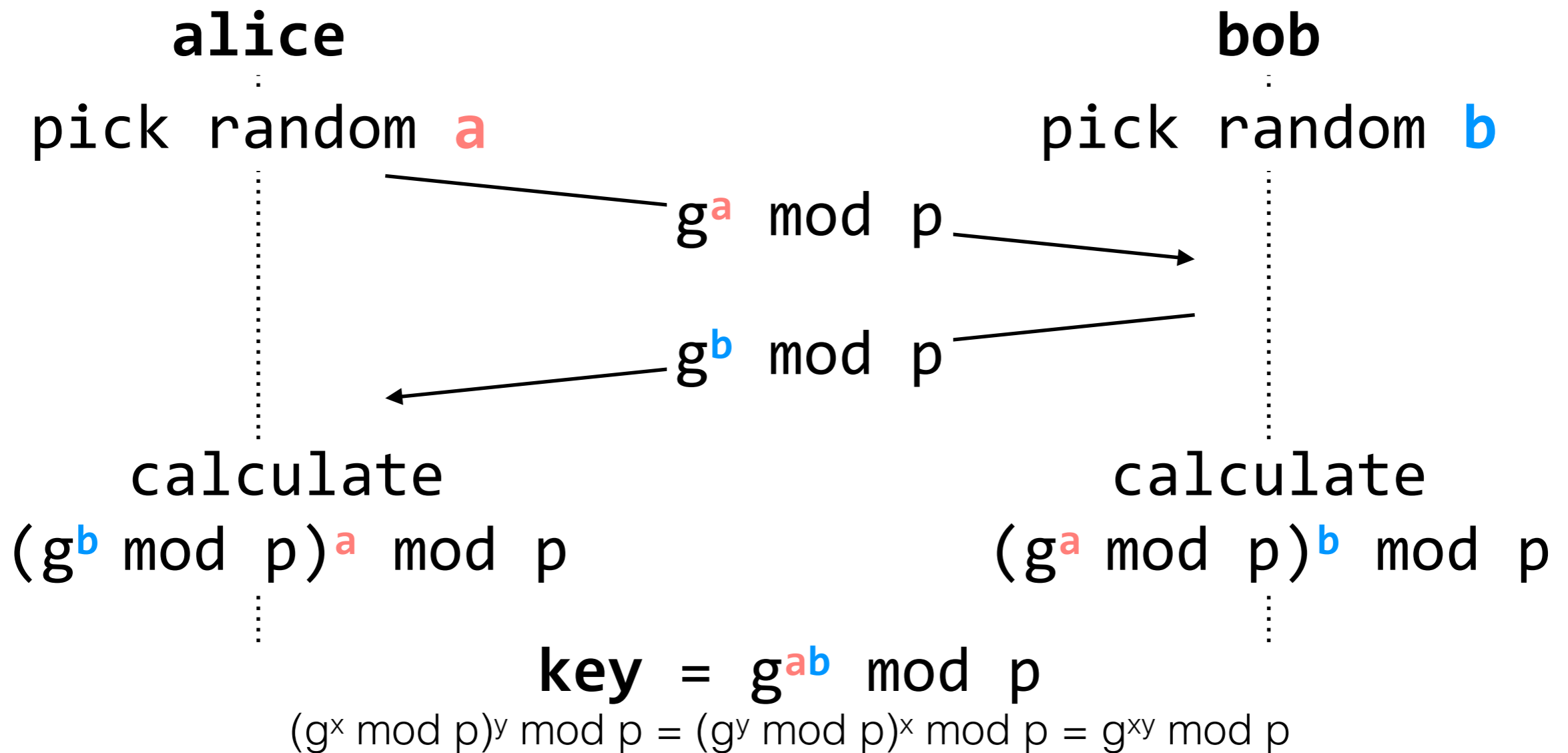


$$m_b \mid \text{seq}_b = \text{decrypt}(k_b, c_b)$$
$$\text{MAC}(k_b, m_b \mid \text{seq}_b) == h_b ?$$

**problem:** how do the parties know the keys?

**known:**  $p$  (prime),  $g$

**property:** given  $g^r \bmod p$ , it is (virtually) impossible to determine  $r$  *even if* you know  $g$  and  $p$





alice

eve

bob

pick random **a**

pick random **e**

pick random **b**

$g^a \text{ mod } p$

$g^b \text{ mod } p$

$g^e \text{ mod } p$

$g^e \text{ mod } p$

$$k_1 = (g^e)^a \text{ mod } p$$

$$k_2 = (g^e)^b \text{ mod } p$$

eve can calculate  
 $k_1$  and  $k_2$

encrypt( $k_1$ ,  $m$ )

decrypt  $m$

encrypt( $k_2$ ,  $m$ )

**problem:** alice and bob don't know they're not communicating directly

# cryptographic signatures

allow users to verify identities using public-key cryptography

## users generate key pairs

the two keys in the pair are related mathematically

{**public\_key**, **secret\_key**}

**sign**(**secret\_key**, message) → **sig**

**verify**(**public\_key**, message, **sig**) → yes/no

**property:** it is (virtually) impossible to compute **sig** without  
**secret\_key**

# TLS handshake

client

server

`ClientHello {version, seqc, session_id, cipher suites, compression func}` →

← `ServerHello {version, seqs, session_id, cipher suite, compression func}`

← `{server certificate, CA certificates}`

← `ServerHelloDone`

client verifies authenticity of server

→ `ClientKeyExchange {encrypt(server_pub_key, pre_master_secret)}`

compute

`master_secret = PRF(pre_master_secret, "master secret", seqc | seqs)`

`key_block = PRF(master_secret, "key expansion", seqc | seqs)`

`= {client_MAC_key,  
server_MAC_key,  
client_encrypt_key,  
server_encrypt_key,  
...}`

→ `Finished {sign(client_MAC_key, encrypt(client_encrypt_key,  
MAC(master_secret, previous_messages)))}`

← `Finished {sign(server_MAC_key, encrypt(server_encrypt_key,  
MAC(master_secret, previous_messages)))}`

- **Secure channels** protect us from adversaries that can observe and tamper with packets in the network.
- Encrypting with **symmetric keys** provides secrecy, and using **MACs** provides integrity. **Diffie-Hellman key exchange** lets us exchange the symmetric key securely.
- To verify identities, we use **public-key cryptography** and cryptographic **signatures**. We often distribute public keys with **certificate authorities**, though this method is not perfect.