2550 Intro to cybersecurity

L13: Signatures

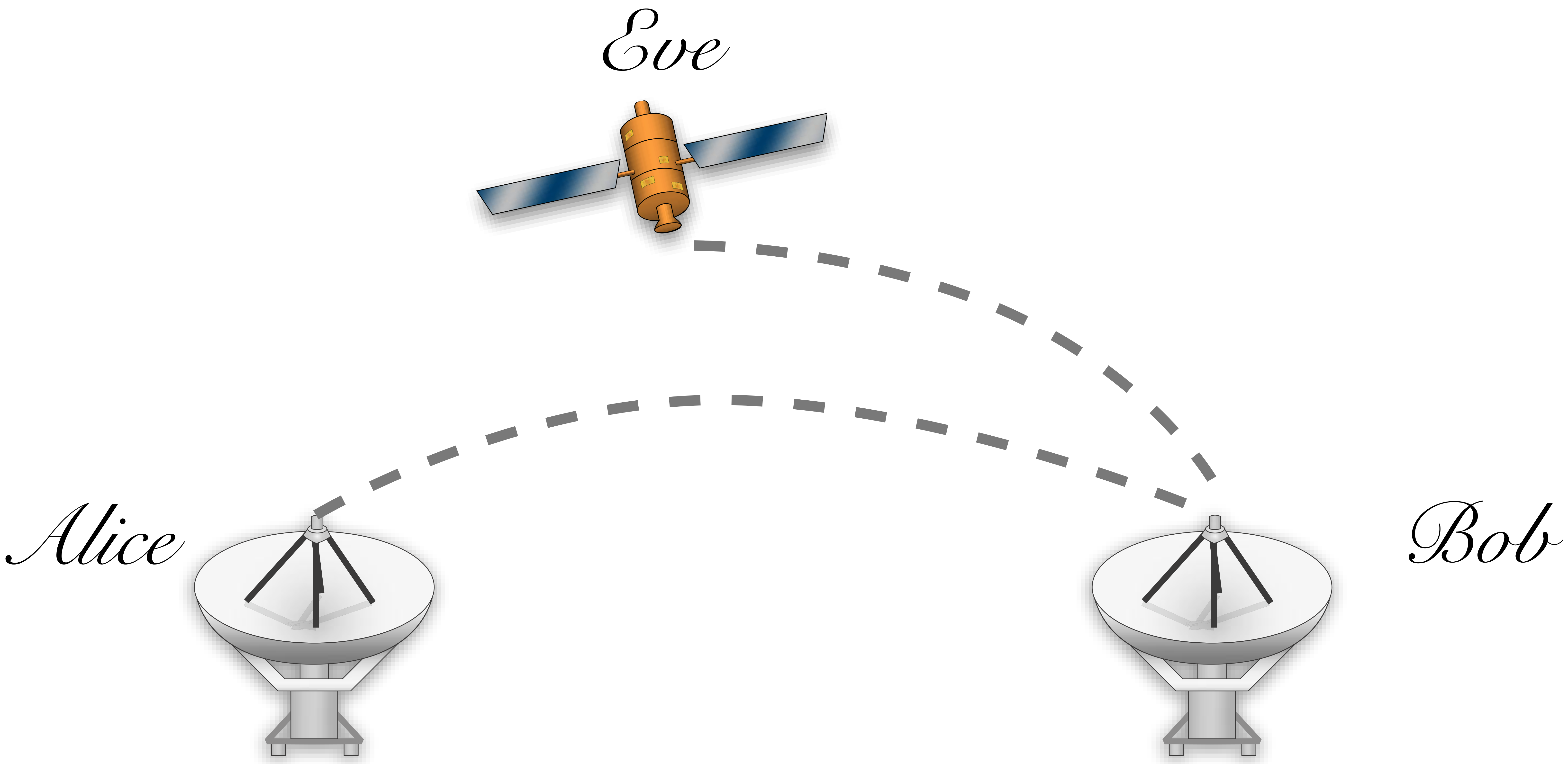
abhi shelat

Recap

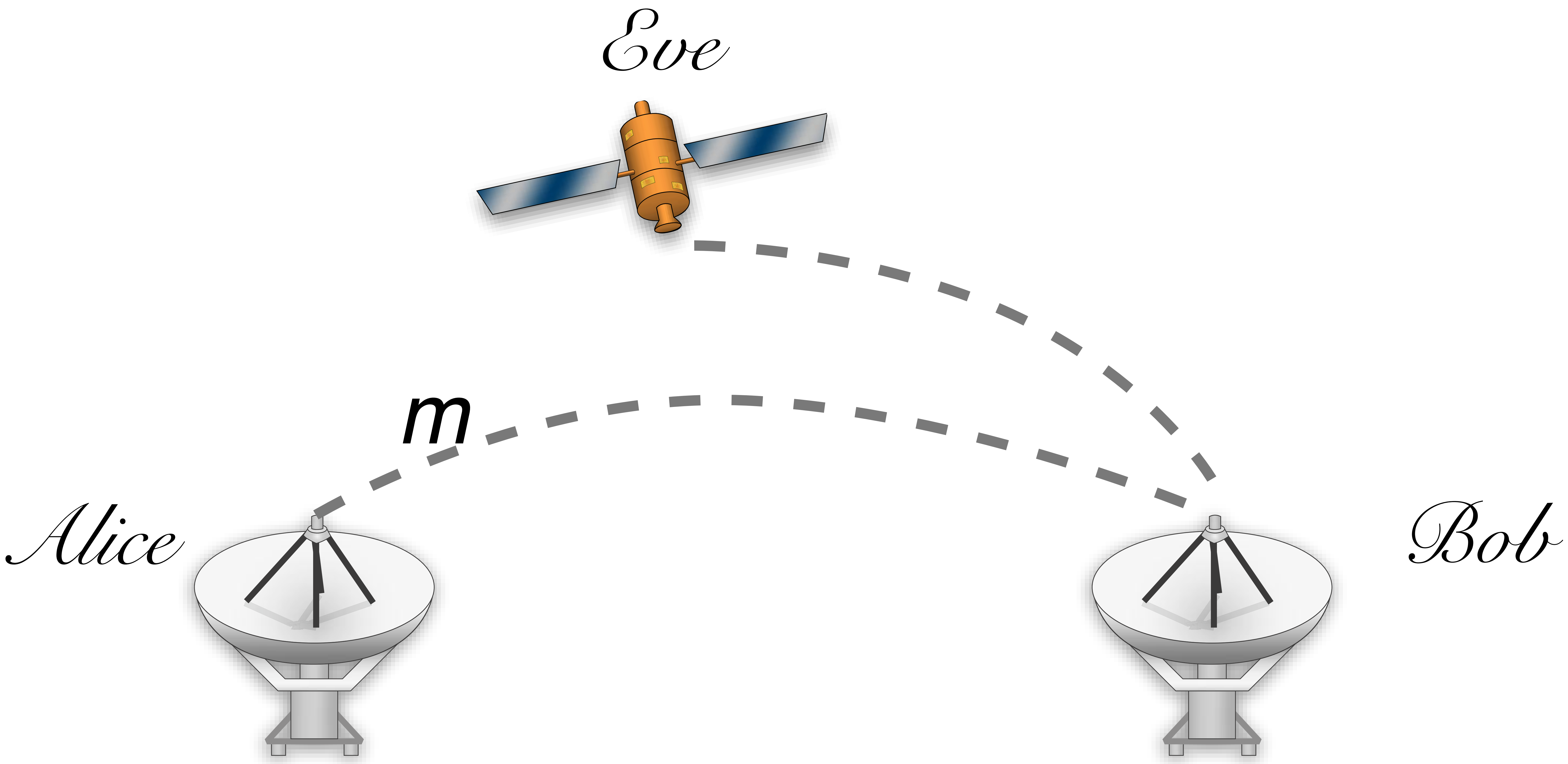
Very old problem

John Hancock

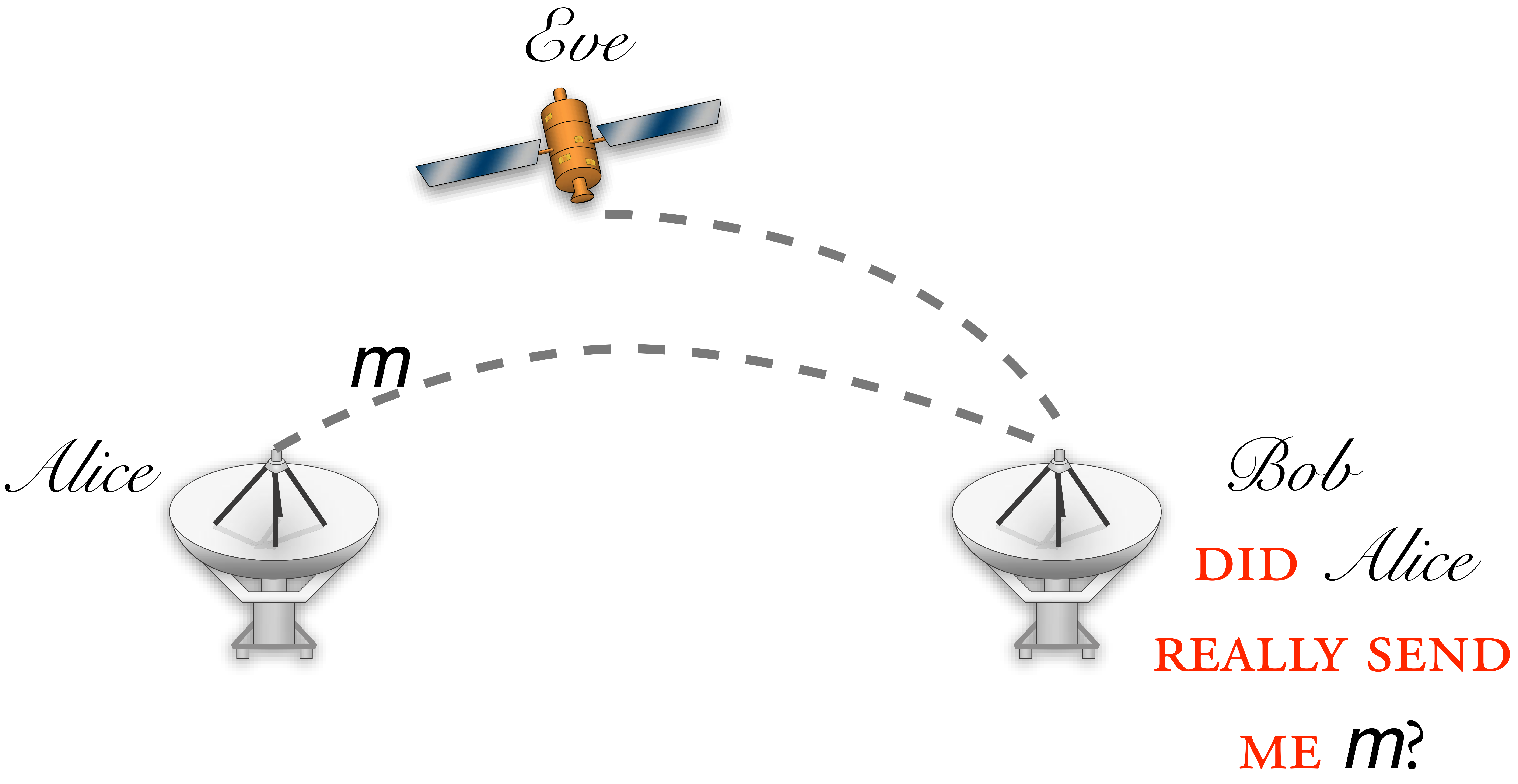
New Problem



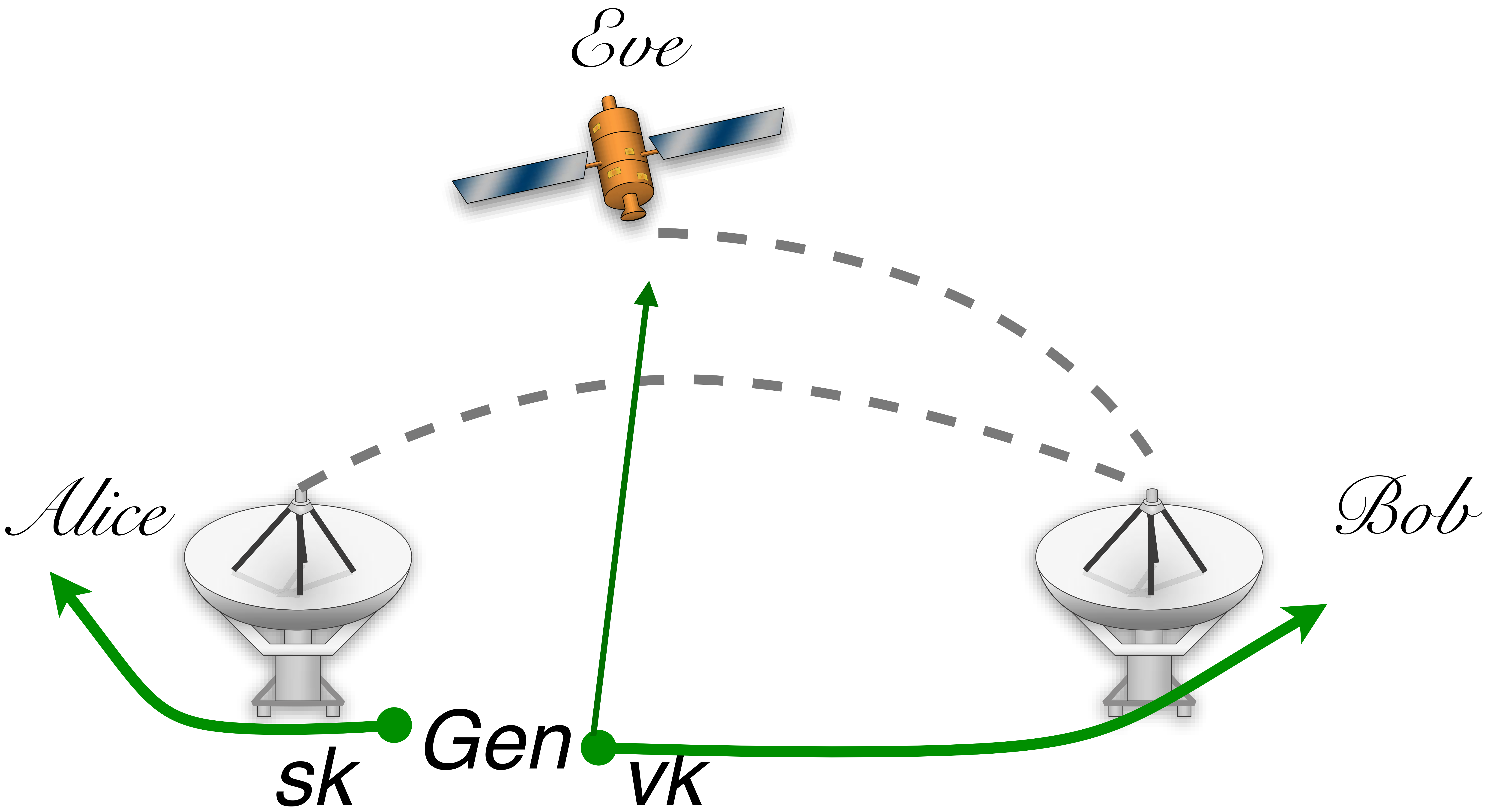
New Problem



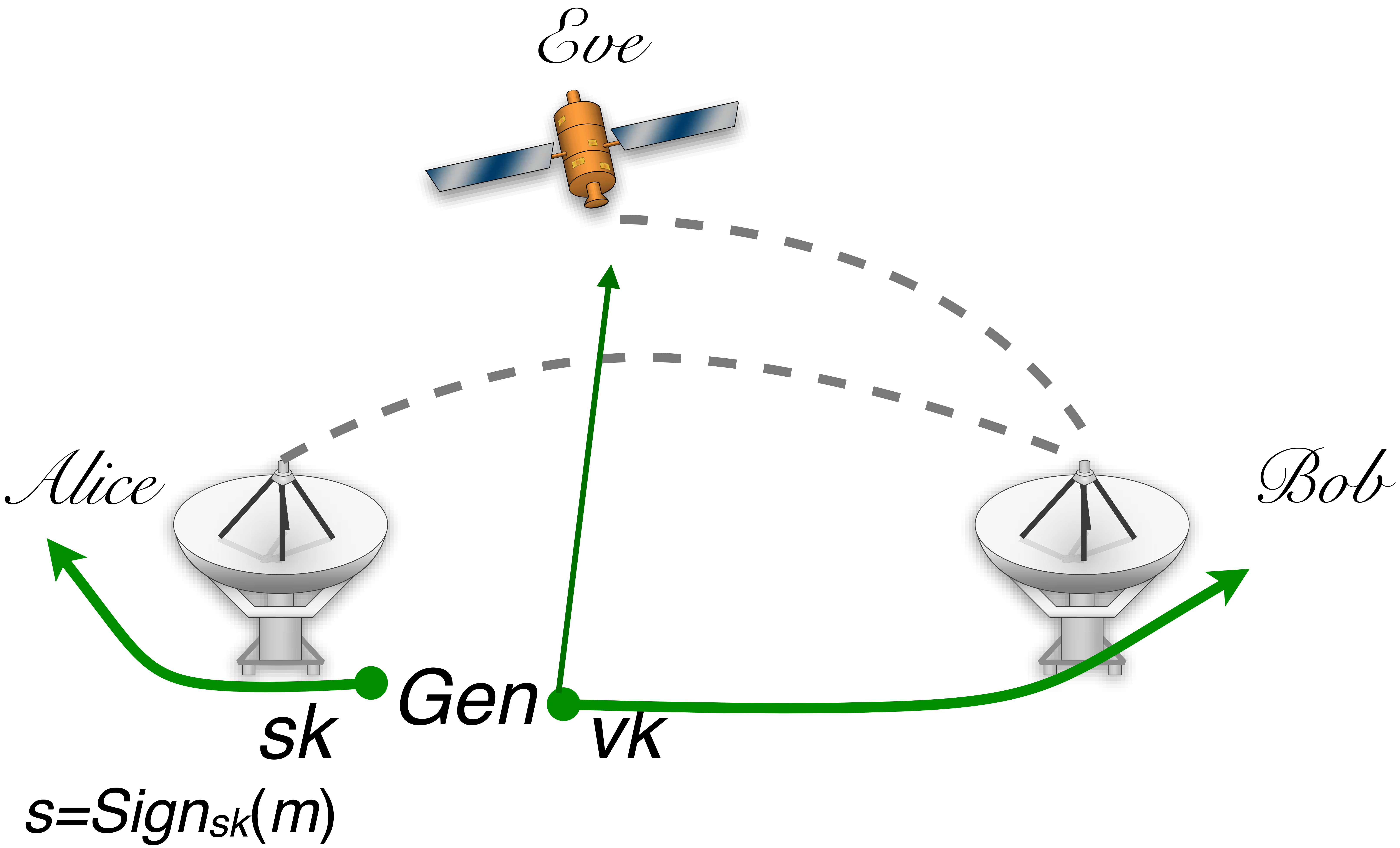
New Problem



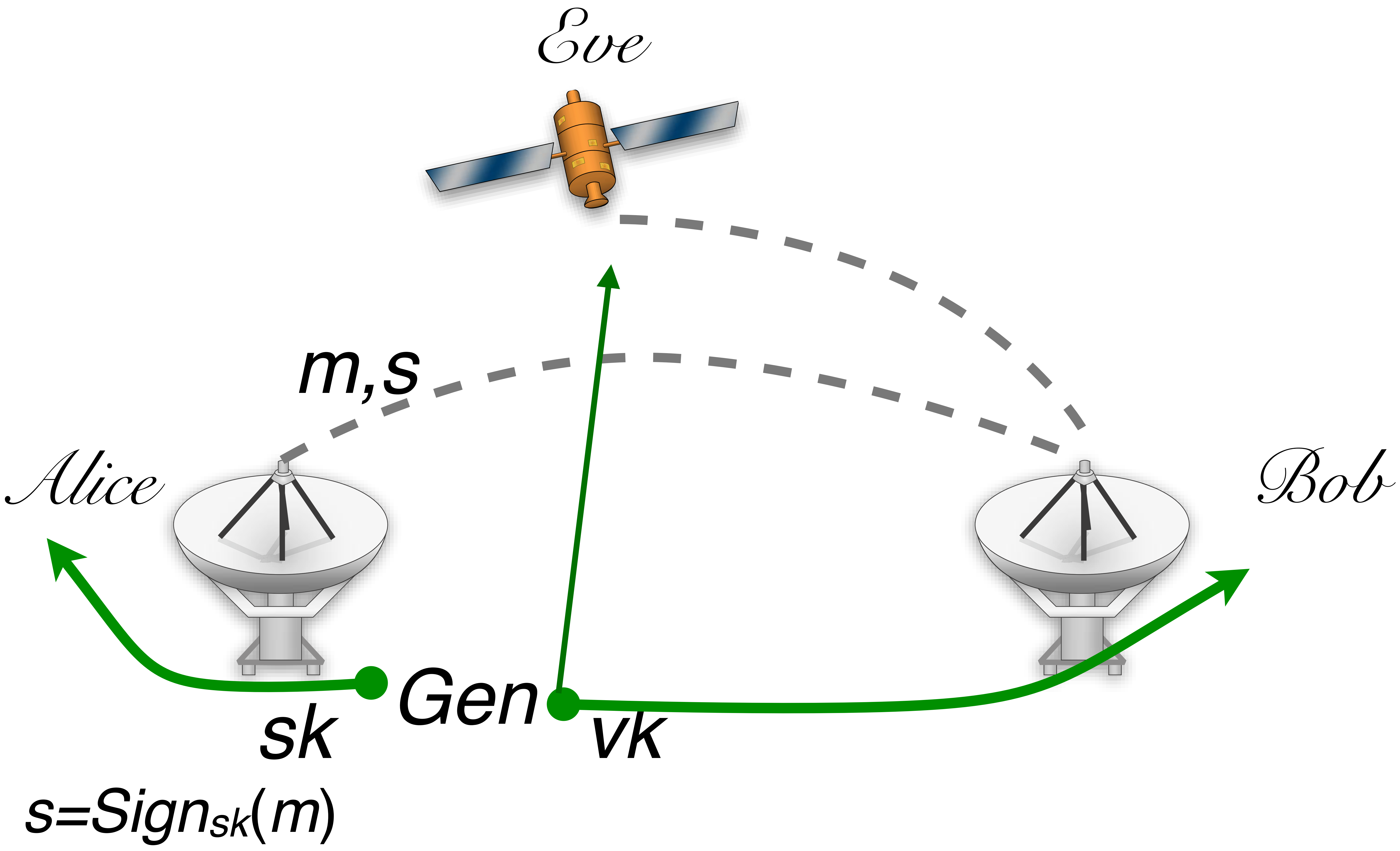
Public key digital signature



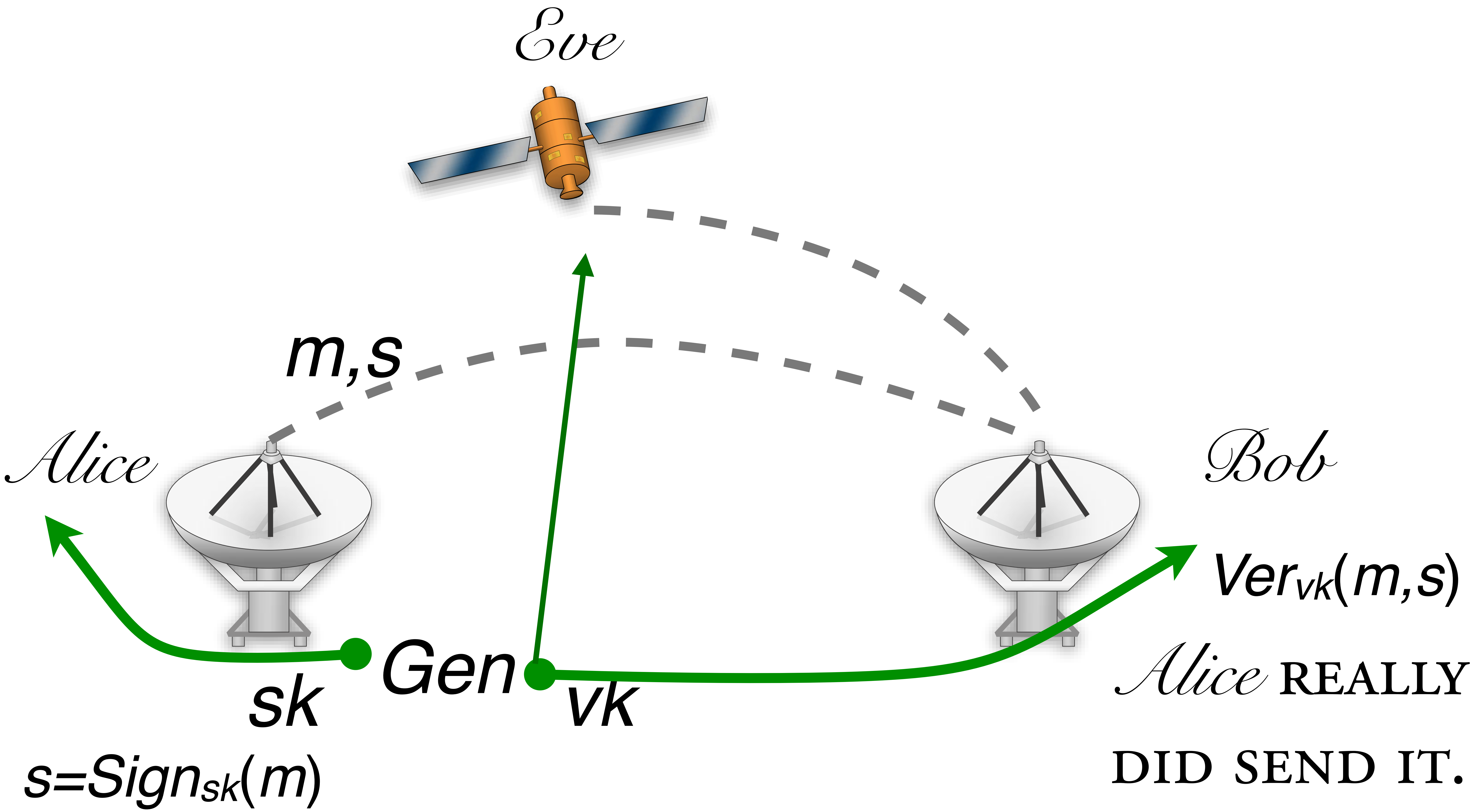
Public key digital signature



Public key digital signature



Public key digital signature



Public key digital signature

MESSAGE SPACE $\{\mathcal{M}\}_n$

$Gen(1^n)$

$Sign_{sk}(m)$

$Ver_{vk}(m,s)$

Public key digital signature

MESSAGE SPACE $\{\mathcal{M}\}_n$

$Gen(1^n)$ GENERATES A KEY PAIR sk, vk

$Sign_{sk}(m)$

$Ver_{vk}(m,s)$

Public key digital signature

MESSAGE SPACE $\{\mathcal{M}\}_n$

Gen(1^n) GENERATES A KEY PAIR *sk,vk*

Sign$_{sk}$(*m*) GENERATES A SIGNATURE *s* FOR $m \in \mathcal{M}_n$

Ver$_{vk}$(*m,s*)

Public key digital signature

MESSAGE SPACE $\{\mathcal{M}\}_n$

Gen(1^n) GENERATES A KEY PAIR *sk,vk*

Sign$_{sk}$(*m*) GENERATES A SIGNATURE *s* FOR $m \in \mathcal{M}_n$

Ver$_{vk}$(*m,s*) ACCEPTS OR REJECTS A MSG,SIG PAIR

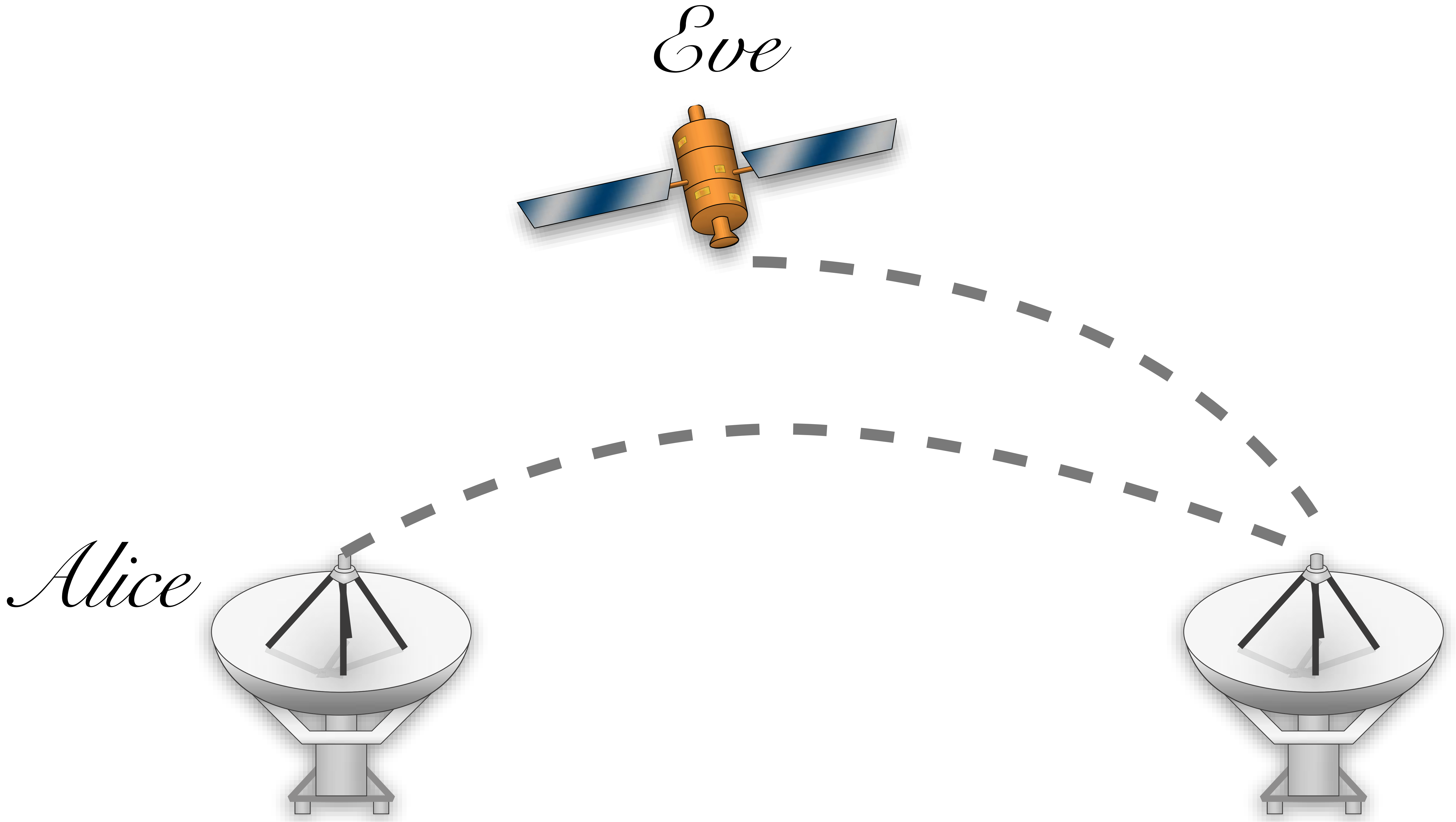
$$\Pr[k \leftarrow Gen(1^n) : Ver_{vk}(m, Sign_{sk}(m)) = 1] = 1$$

existential unforgability

"EVEN WHEN GIVEN A SIGNING ORACLE,

AN ADVERSARY CANNOT FORGE A SIGNATURE FOR

ANY MESSAGE OF ITS CHOOSING"

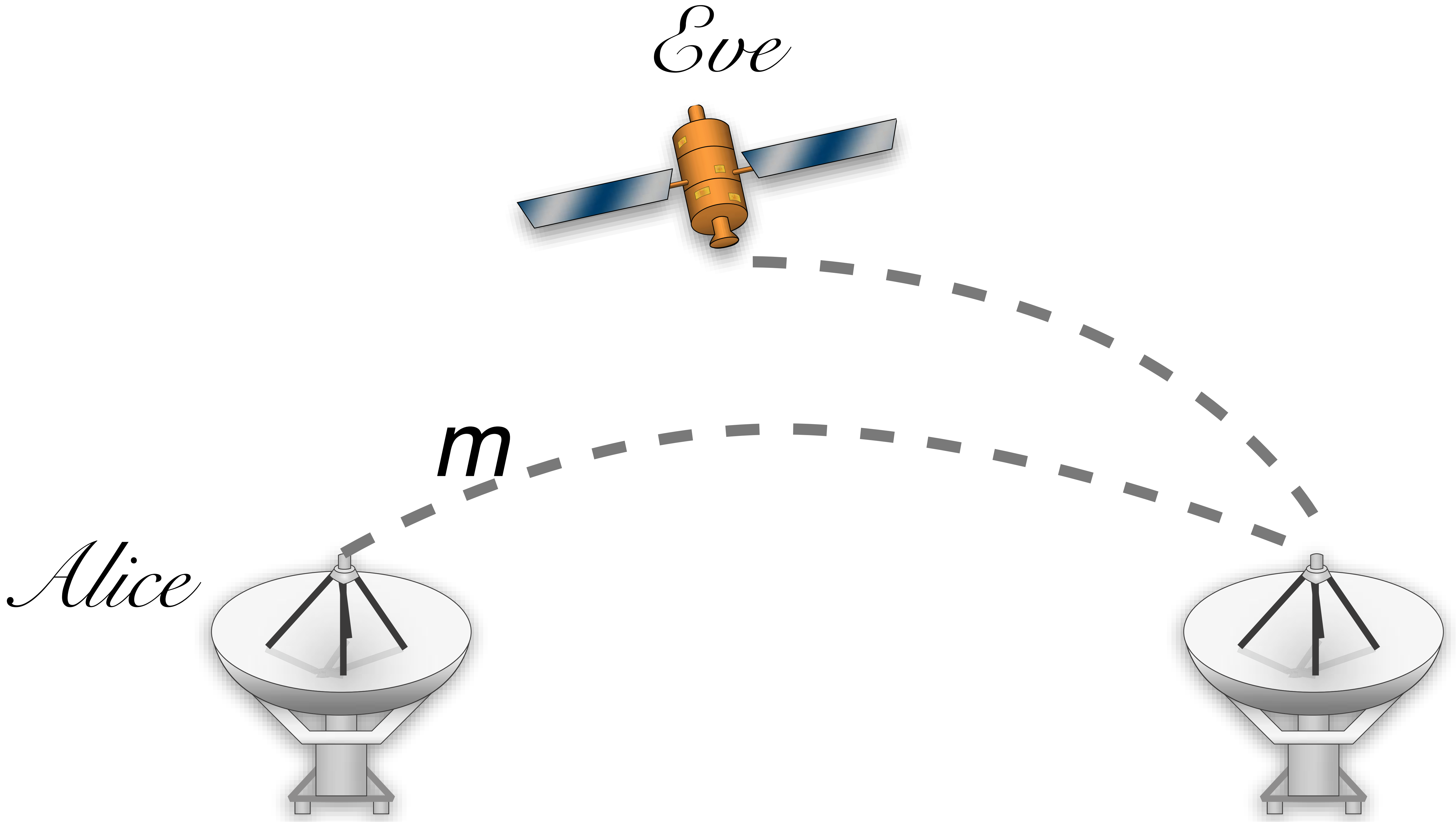


existential unforgability

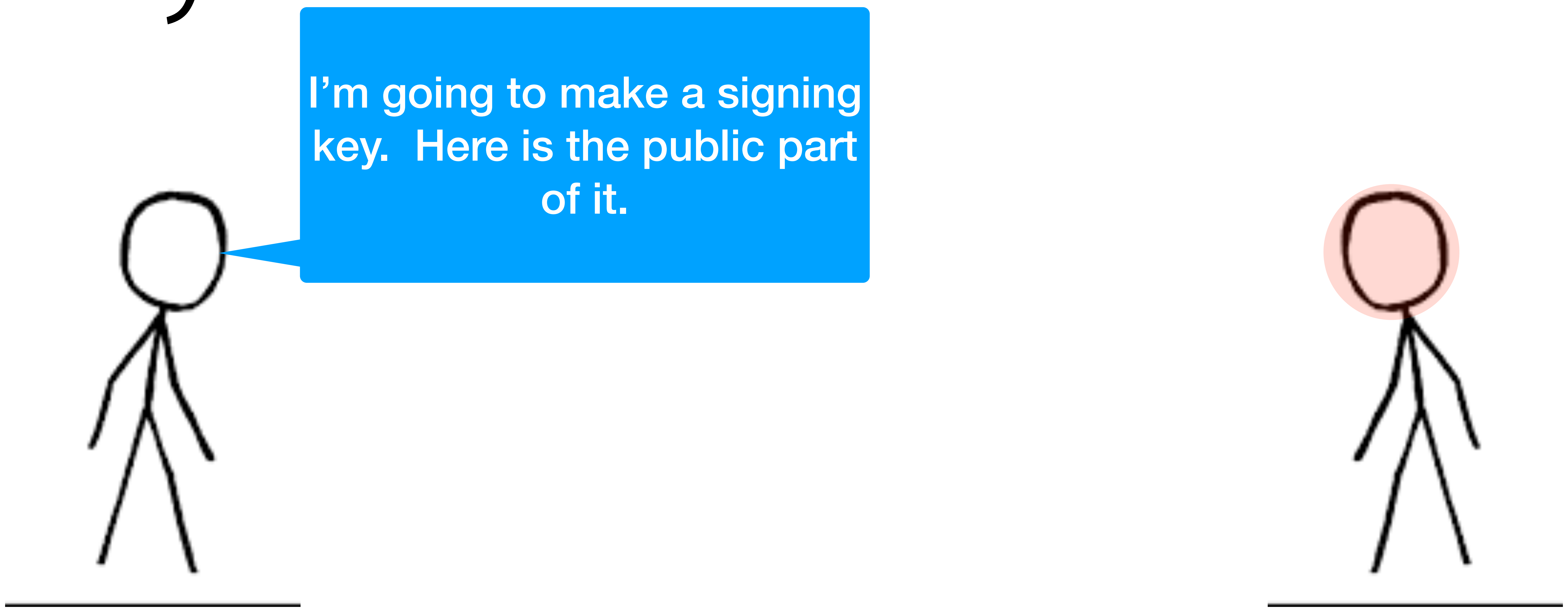
"EVEN WHEN GIVEN A SIGNING ORACLE,

AN ADVERSARY CANNOT FORGE A SIGNATURE FOR

ANY MESSAGE OF ITS CHOOSING"



Signature security



$$(vk, sk) \leftarrow \mathrm{Gen}(1^n)$$

Signature security

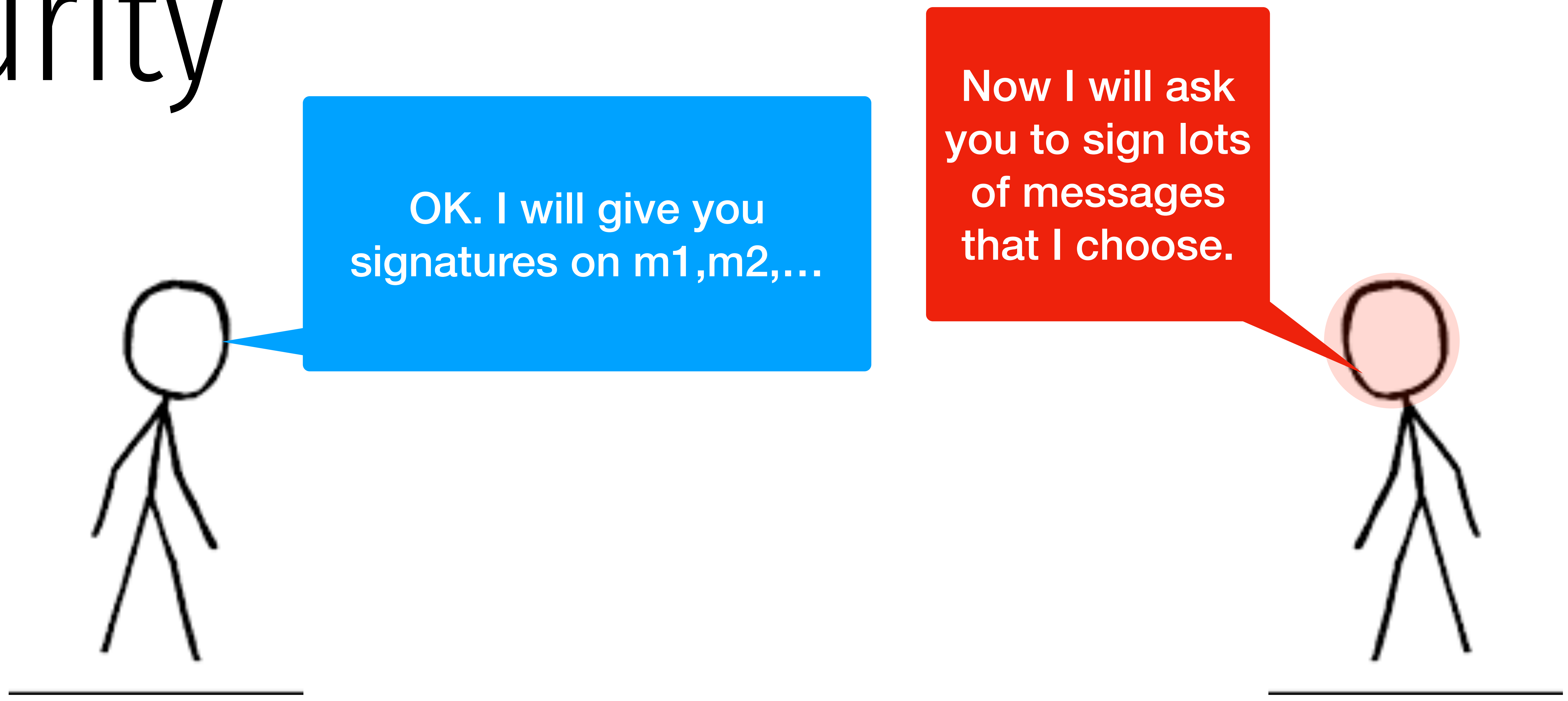
$(vk, sk) \leftarrow \text{Gen}(1^n)$

Now I will ask you to sign lots of messages that I choose.

$m_0, m_1, \ldots$

vk

Signature security



$$(vk, sk) \leftarrow \text{Gen}(1^n)$$

$$vk$$

$$s_i \leftarrow \mathsf{Sign}_{sk}(m_i)$$

Signature security

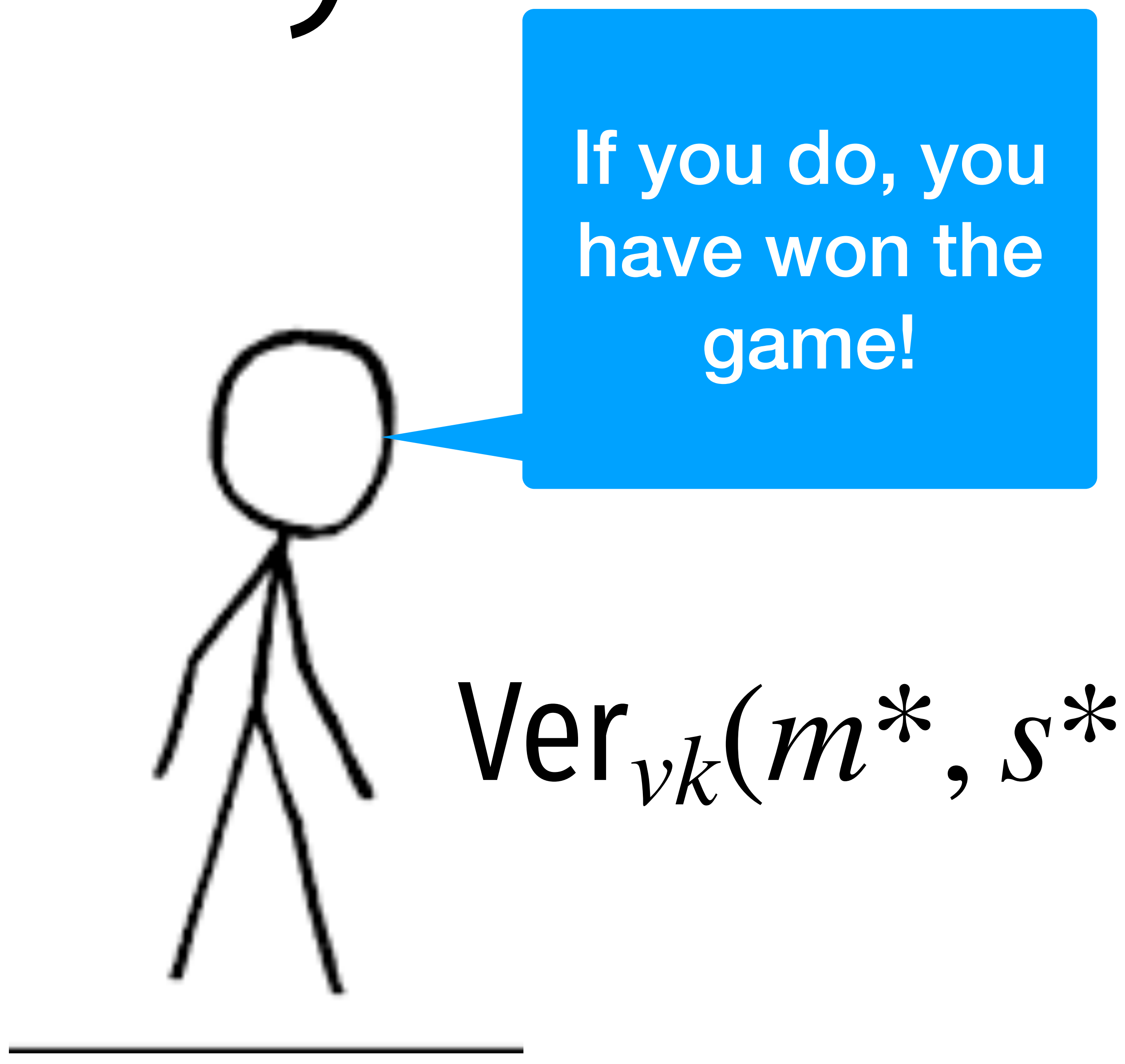
$(vk, sk) \leftarrow \mathrm{Gen}(1^n)$

$s_i \leftarrow \mathsf{Sign}_{sk}(m_i)$

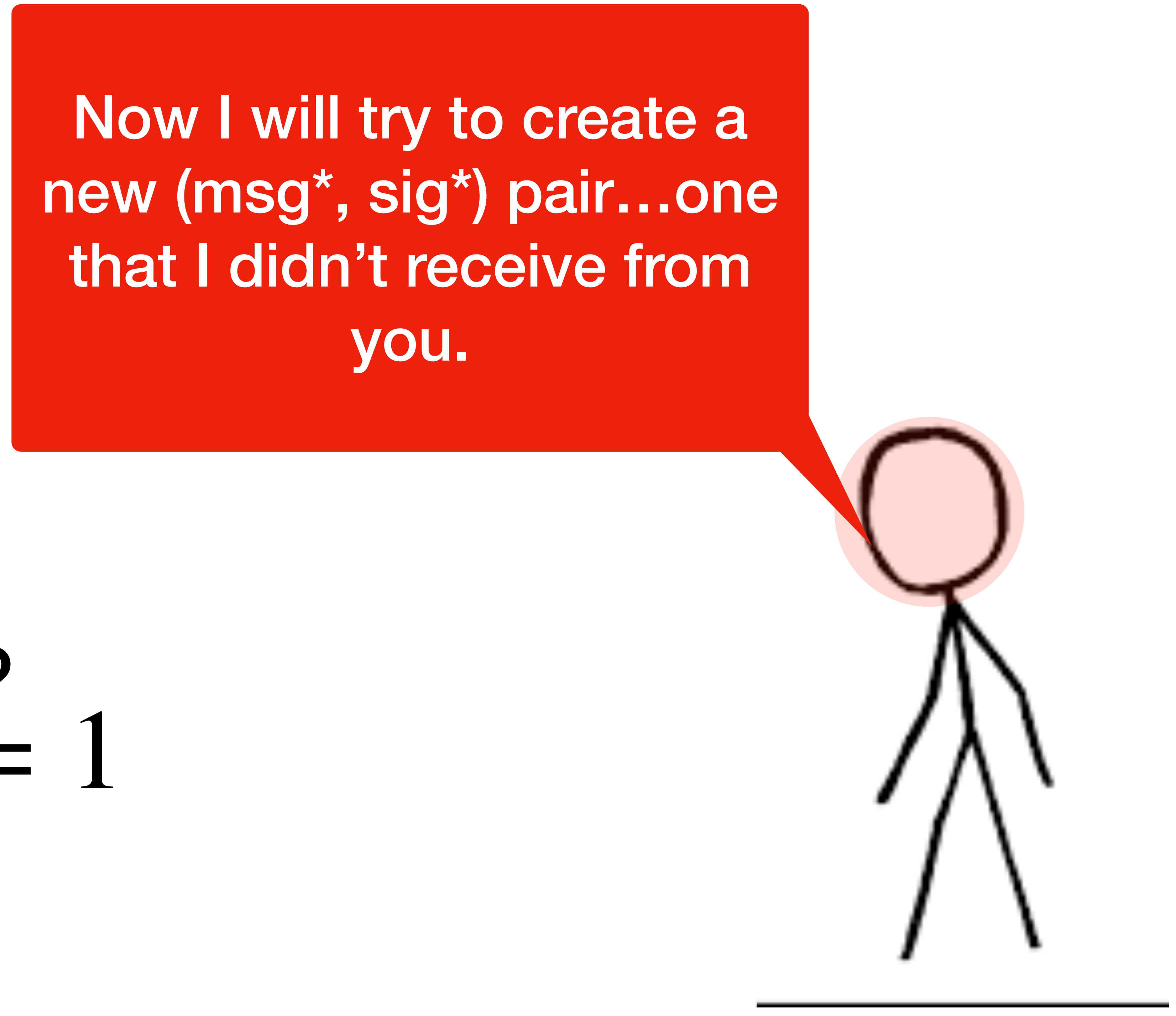
vk

$s_1, s_2, \ldots$

Signature security



$$\mathsf{Ver}_{vk}(m^*, s^*) \stackrel{?}{=} 1$$



FOR ALL NON-UNIFORM PPT *A*

$$\Pr\left[\begin{array}{l}(vk, sk) \leftarrow Gen(1^n); (m, s) \leftarrow A^{Sign_{sk}(\cdot)} : \\ \quad Ver_{vk}(m, s) = 1 \\ \quad \text{AND } A \text{ DIDN'T QUERY } m\end{array}\right] < \mu(n)$$

Textbook RSA Signatures (insecure)

Pick N = p*q where p,q are primes.

Pick e,d such that $e \cdot d = 1 \bmod \phi(N)$

Textbook RSA Signatures (insecure)

Pick N = p*q where p,q are primes.

Pick e,d such that $e \cdot d = 1 \bmod \phi(N)$

Sign((sk=d, N) m):

Compute the signature: $\sigma \leftarrow m^d \bmod N$

Verify((pk=e, N), σ, m):

$$m \stackrel{?}{=} \sigma^e \bmod N$$

Textbook RSA (insecure) example

Lets pick a key N = 443 * 919 = 407177.

Lets say e = 65537. What is d ?

Textbook RSA (insecure) example

Lets pick a key N = 443 * 919 = 407177.

Lets say e = 65537. What is d ?

Sign the message m = “22” = 0x3232 = 12850.

Textbook RSA (insecure) example

Lets pick a key N = 443 * 919 = 407177.

Lets say e = 65537. What is d ?

Sign the message m = “22” = 0x3232 = 12850.

sig = 84760.

Textbook RSA (insecure) example

Lets pick a key N = 443 * 919 = 407177.

Lets say e = 65537. What is d ?

Sign the message m = “22” = 0x3232 = 12850.

sig = 84760.

Verify the signature (“22”, 84760) :

Textbook RSA Signatures (insecure)

Pick N = p*q where p,q are primes.

Pick e,d such that $e \cdot d = 1 \bmod \phi(N)$

Sign((sk=d, N) m):

Compute the signature: $\sigma \leftarrow m^d \bmod N$

Verify((pk=e, N), σ, m): $m \stackrel{?}{=} \sigma^e \bmod N$

Why is this scheme insecure?

Textbook RSA Signatures (insecure)

Pick N = p*q where p,q are primes.

Pick e,d such that $e \cdot d = 1 \bmod \phi(N)$

Sign((sk=d, N) m):

Compute the signature: $\sigma \leftarrow m^d \bmod N$

Verify((pk=e, N), σ, m): $m \stackrel{?}{=} \sigma^e \bmod N$

Why is this scheme insecure?

Given the signature pair (“22” = 12850, 84760) ,
what is the signature on 12850 * 12850 = 165122500 ?

RSA Signatures (PKCSv1.5)

(Randomized padding to prevent basic forgery attacks. Widely used, but first full security proof was written in 2018.)

Sign((sk, N) m):

Compute the padding: $z \leftarrow 00 \cdot 01 \cdot FF \cdots FF \cdot 00 \cdot \mathsf{ID}_H \cdot H(m)$

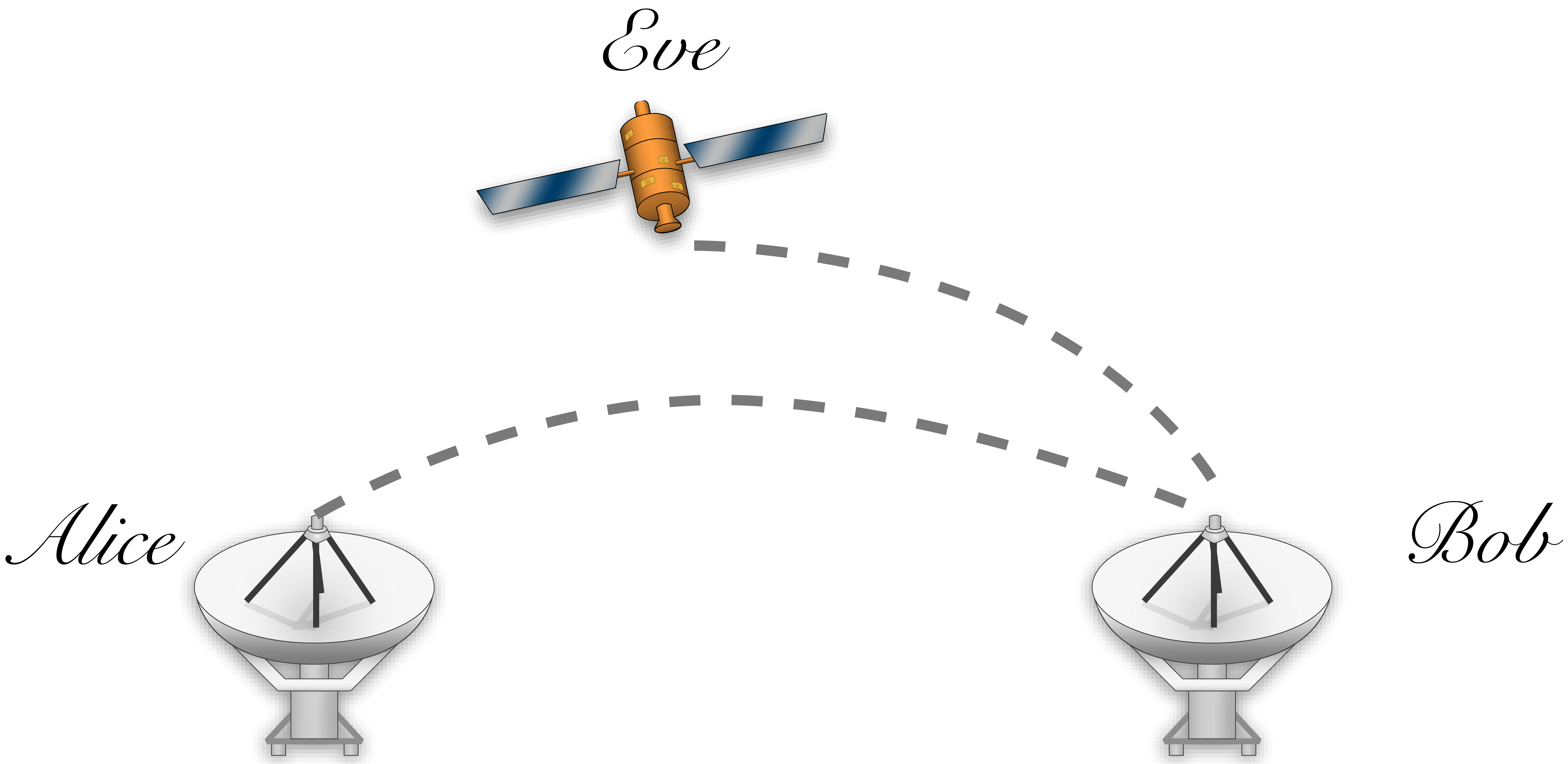
Compute the signature: $\sigma \leftarrow z^{sk} \bmod N$

Speed

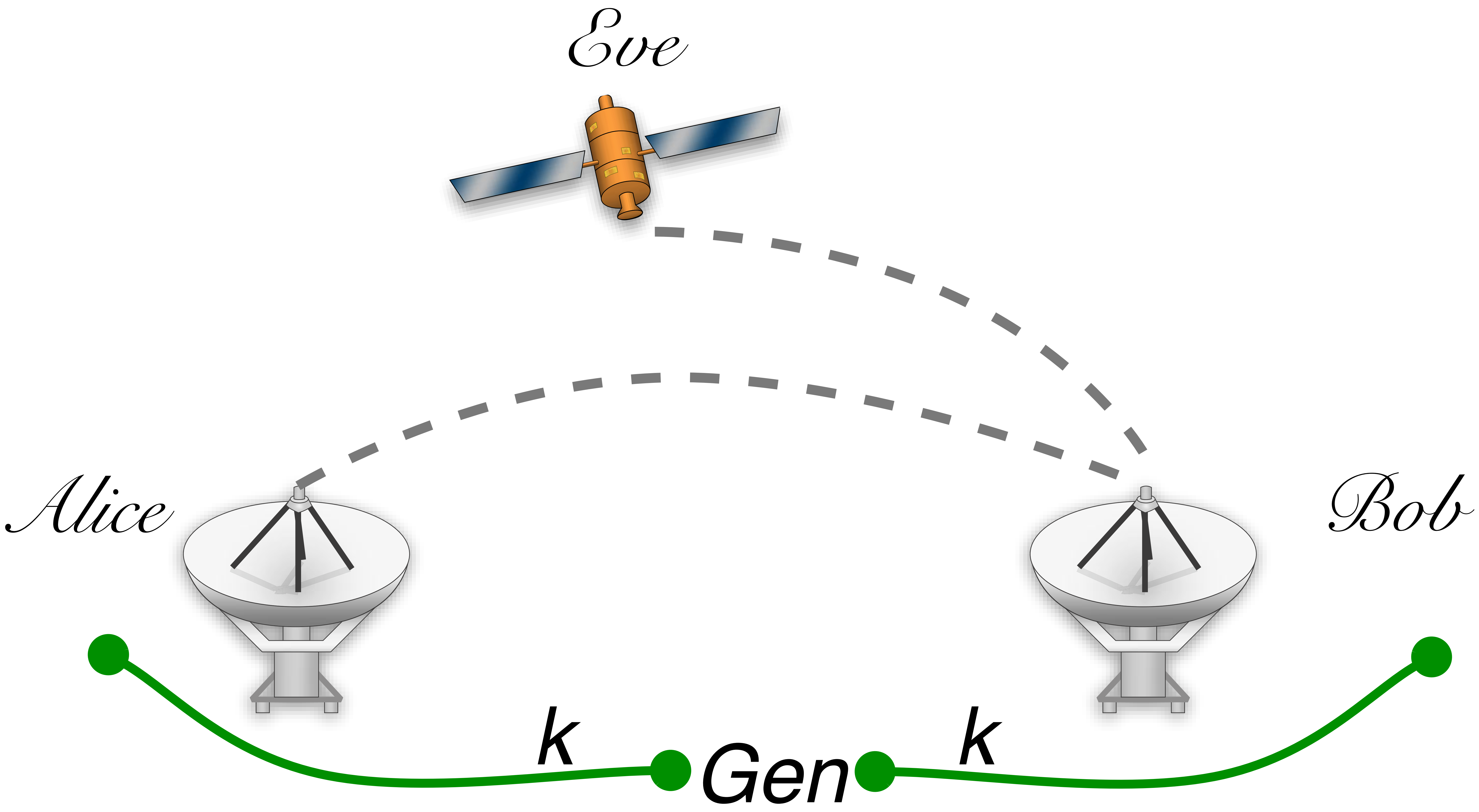
openssl speed rsa dsa ecdsa

```
Doing 1024 bits private rsa's for 10s: 86688 1024 bits private RSA's in 9.99s
Doing 1024 bits public  rsa's for 10s: 1341152 1024 bits public RSA's in 10.00s
Doing 2048 bits private rsa's for 10s: 13154 2048 bits private RSA's in 9.99s
Doing 2048 bits public  rsa's for 10s: 437080 2048 bits public RSA's in 10.00s
Doing 3072 bits private rsa's for 10s: 4243 3072 bits private RSA's in 10.00s
Doing 3072 bits public  rsa's for 10s: 211605 3072 bits public RSA's in 10.00s
Doing 4096 bits private rsa's for 10s: 1845 4096 bits private RSA's in 9.99s
Doing 4096 bits public  rsa's for 10s: 125130 4096 bits public RSA's in 9.99s

Doing 1024 bits sign   dsa's for 10s: 74467 1024 bits DSA signs in 9.95s
Doing 1024 bits verify dsa's for 10s: 95863 1024 bits DSA verify in 9.99s
Doing 2048 bits sign   dsa's for 10s: 30197 2048 bits DSA signs in 9.97s
Doing 2048 bits verify dsa's for 10s: 33802 2048 bits DSA verify in 10.00s

Doing 256 bits sign   ecdsa's for 10s: 339010 256 bits ECDSA signs in 9.89s
Doing 256 bits verify ecdsa's for 10s: 115106 256 bits ECDSA verify in 10.00s
Doing 384 bits sign   ecdsa's for 10s: 7773 384 bits ECDSA signs in 9.98s
Doing 384 bits verify ecdsa's for 10s: 10066 384 bits ECDSA verify in 10.00s
Doing 521 bits sign   ecdsa's for 10s: 25316 521 bits ECDSA signs in 9.98s
Doing 521 bits verify ecdsa's for 10s: 12896 521 bits ECDSA verify in 9.99s
Doing 283 bits sign   ecdsa's for 10s: 13860 283 bits ECDSA signs in 9.98s
Doing 283 bits verify ecdsa's for 10s: 7028 283 bits ECDSA verify in 9.99s
Doing 409 bits sign   ecdsa's for 10s: 8441 409 bits ECDSA signs in 9.99s
Doing 409 bits verify ecdsa's for 10s: 4309 409 bits ECDSA verify in 9.98s
```

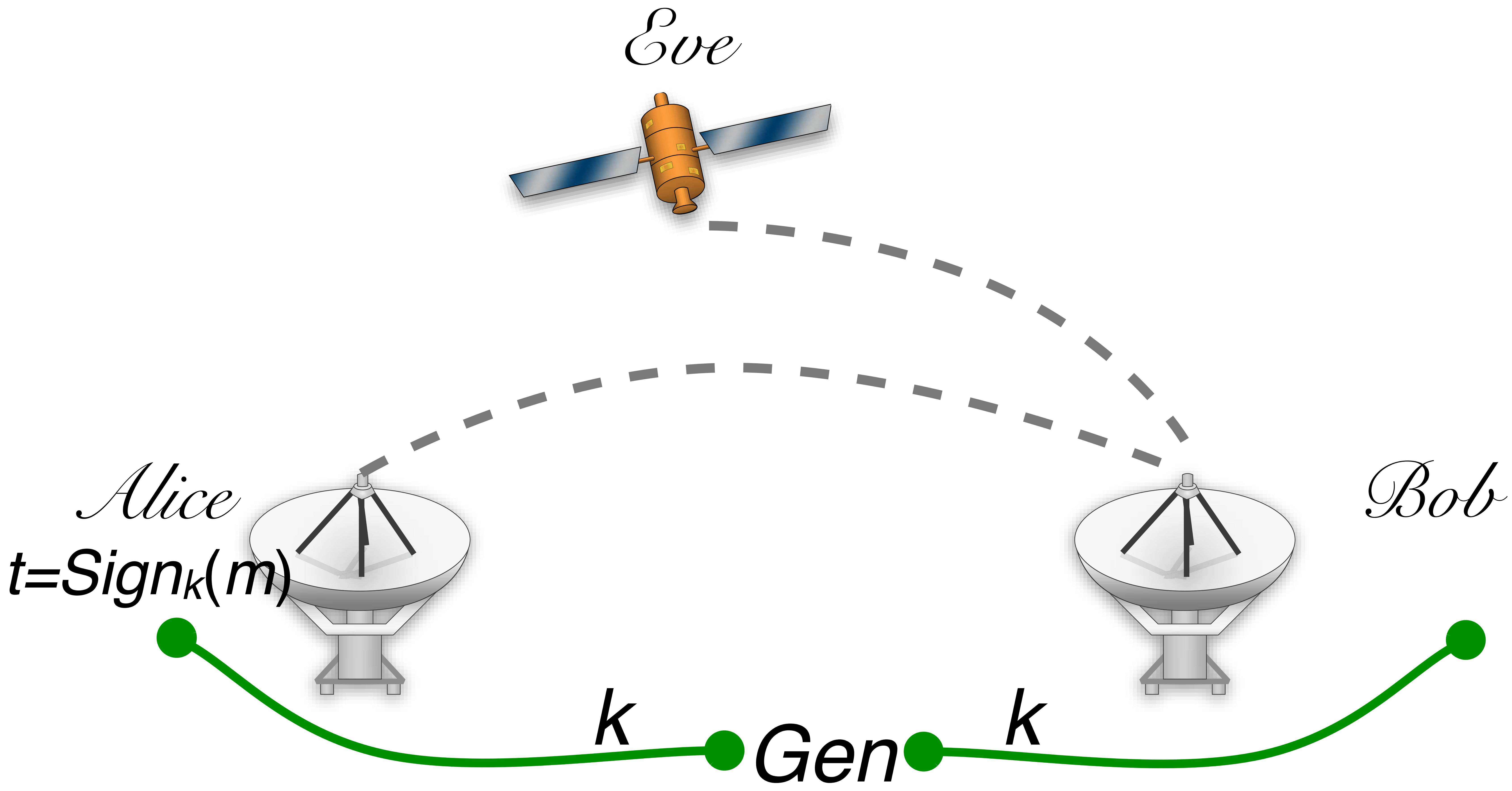
Message Authentication codes



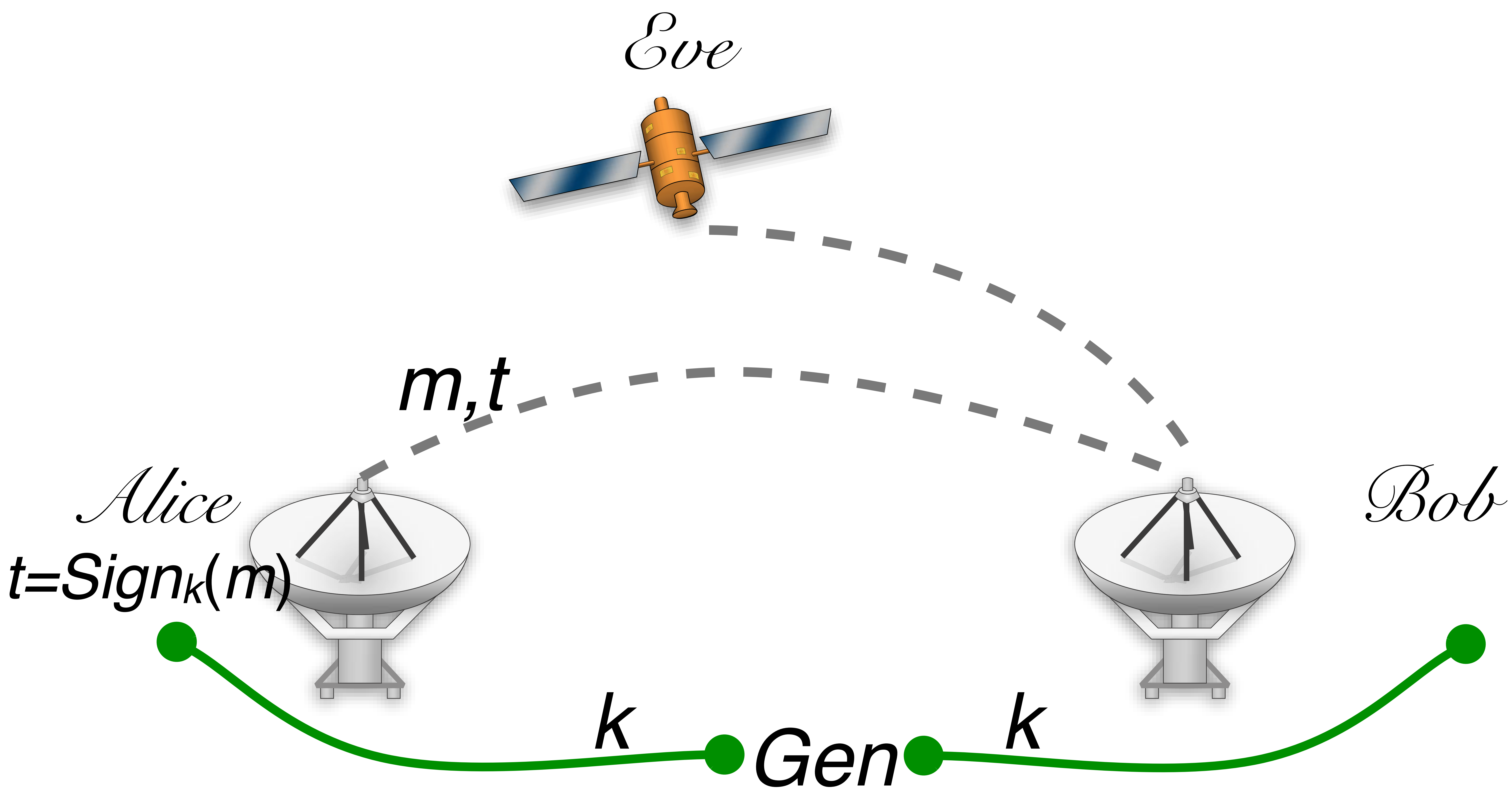
Message Authentication codes



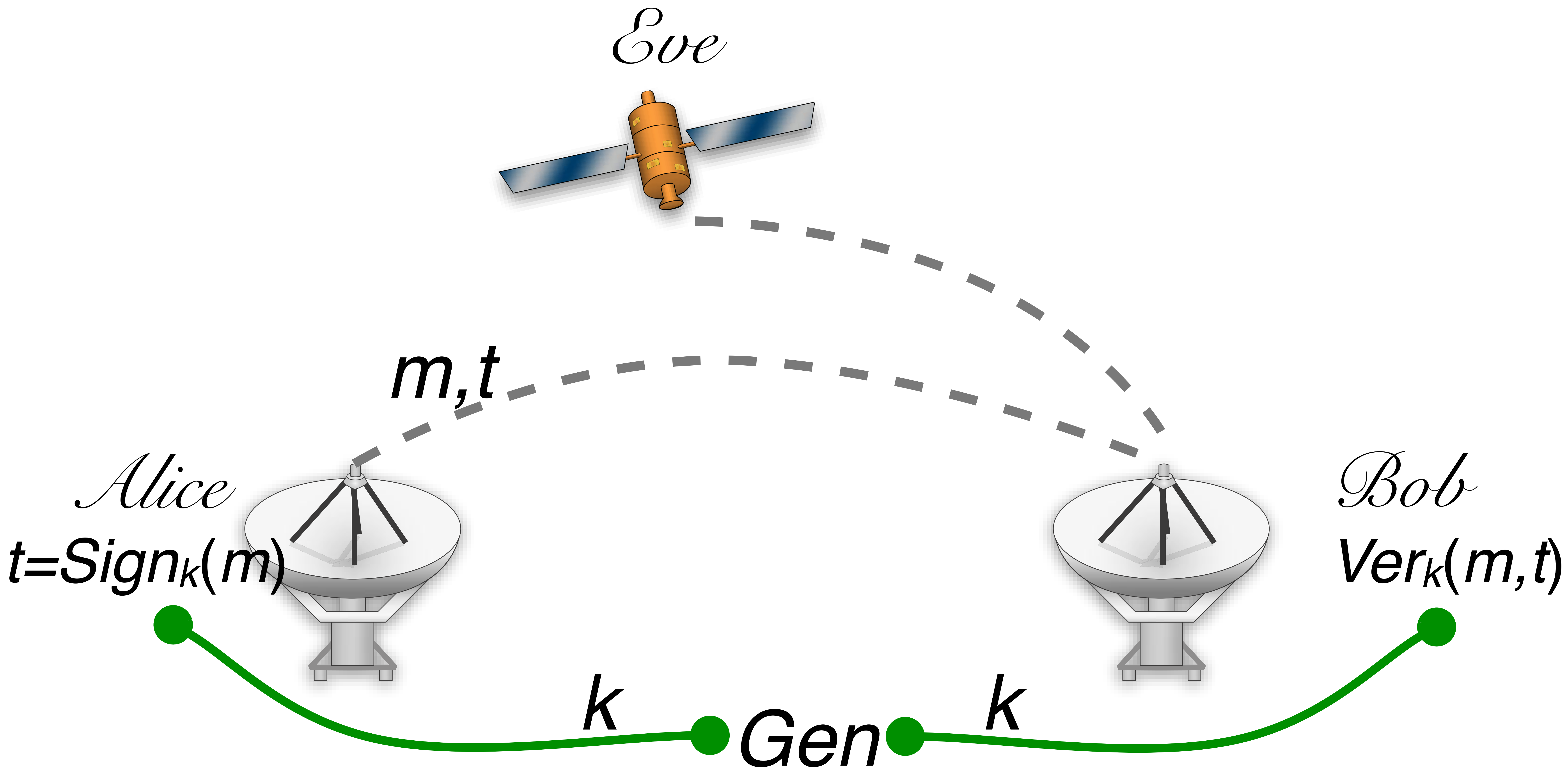
Message Authentication codes



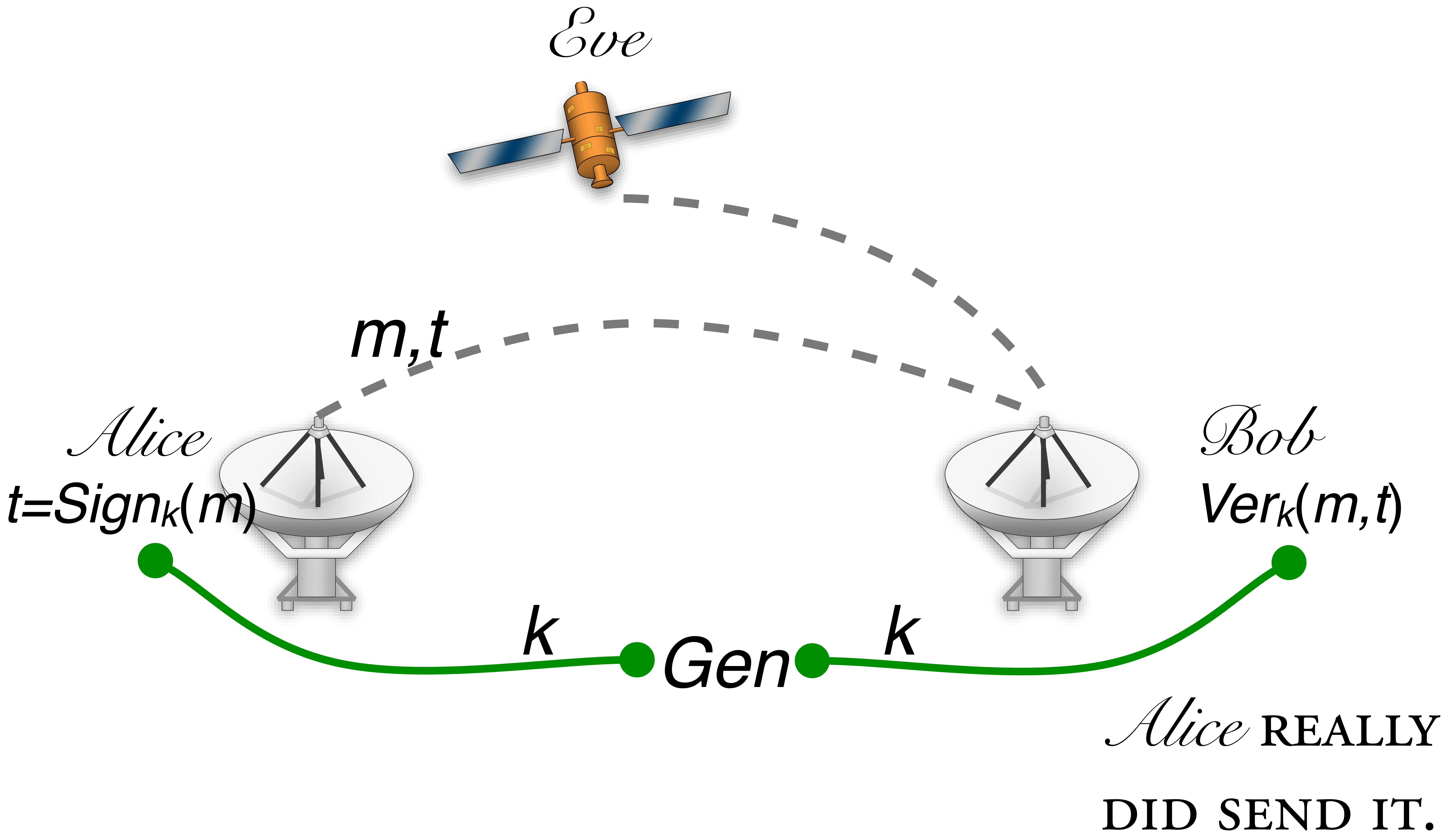
Message Authentication codes



Message Authentication codes



Message Authentication codes



Construction of a MAC

$Gen(1^n)$:

$Sign_k(m)$:

$Ver_k(m,t)$:

Construction of a MAC

LET $\{F_k\}$ BE A PRF FAMILY LIKE AES

Gen(1^n):

Sign$_k$(*m*):

Ver$_k$(*m*,*t*):

Construction of a MAC

LET $\{F_k\}$ BE A PRF FAMILY LIKE AES

Gen(1ⁿ): $k \leftarrow U_n$

Sign$_k$(m):

Ver$_k$(m,t):

Construction of a MAC

LET $\{F_k\}$ BE A PRF FAMILY LIKE AES

Gen(1^n): $k \leftarrow U_n$

Sign_k(m): $t \leftarrow F_k(m)$

Ver_k(m,t):

Construction of a MAC

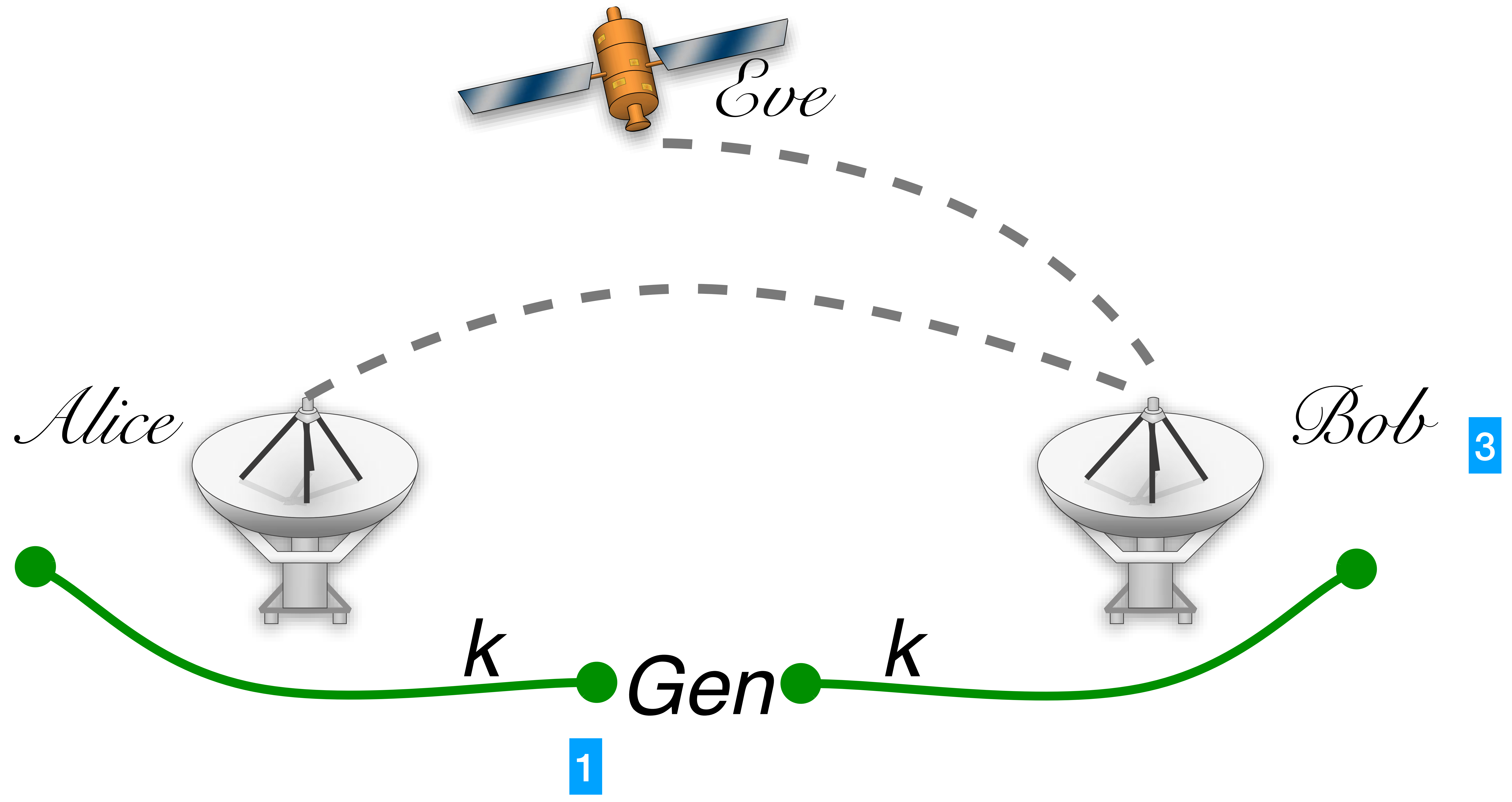
LET $\{F_k\}$ BE A PRF FAMILY LIKE AES

Gen(1^n): $k \leftarrow U_n$

Sign$_k$(m): $t \leftarrow F_k(m)$

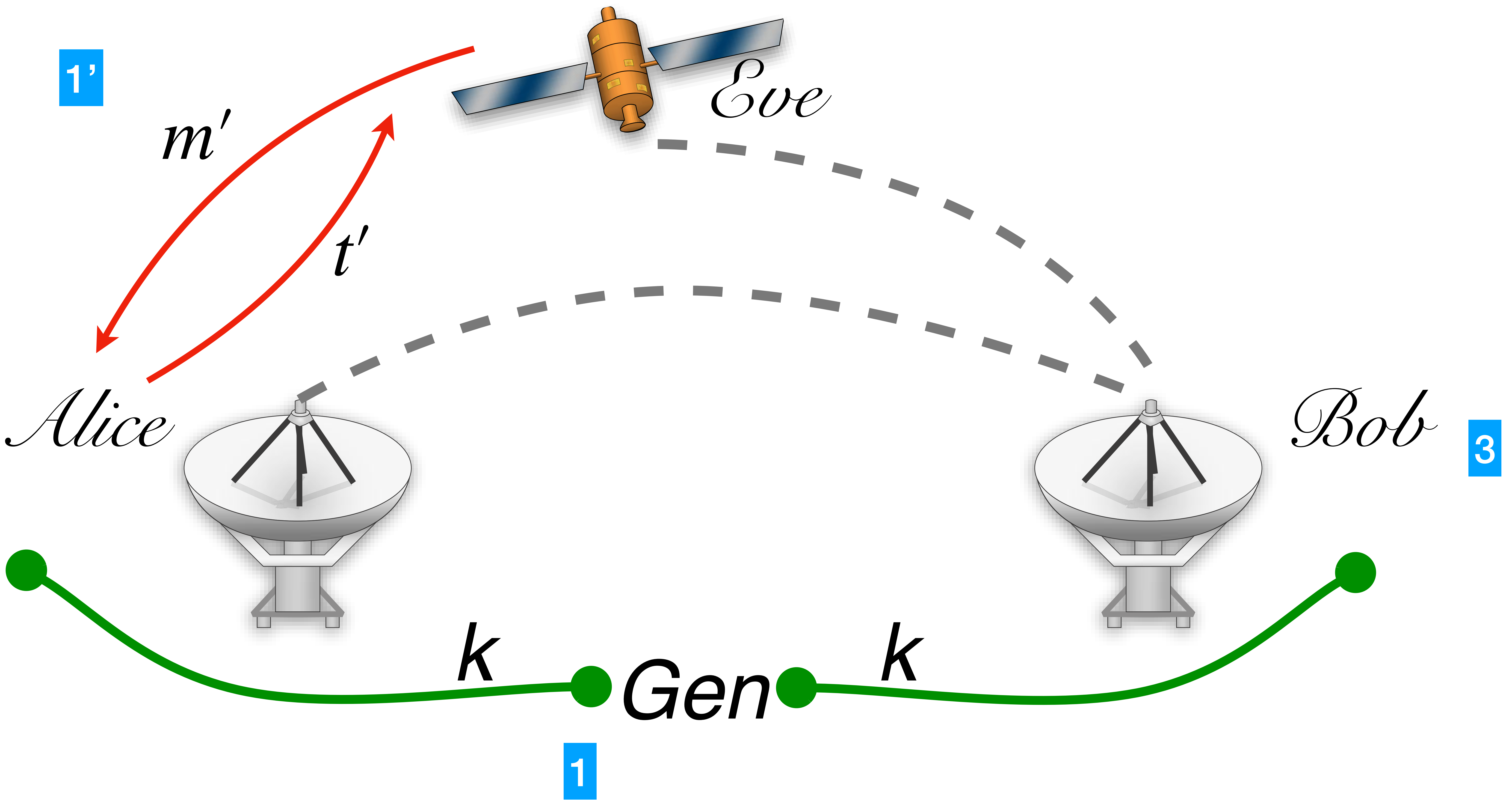
Ver$_k$(m,t): ACCEPT IF $t \stackrel{?}{=} F_k(m)$

Security for a MAC (similar to Signature)

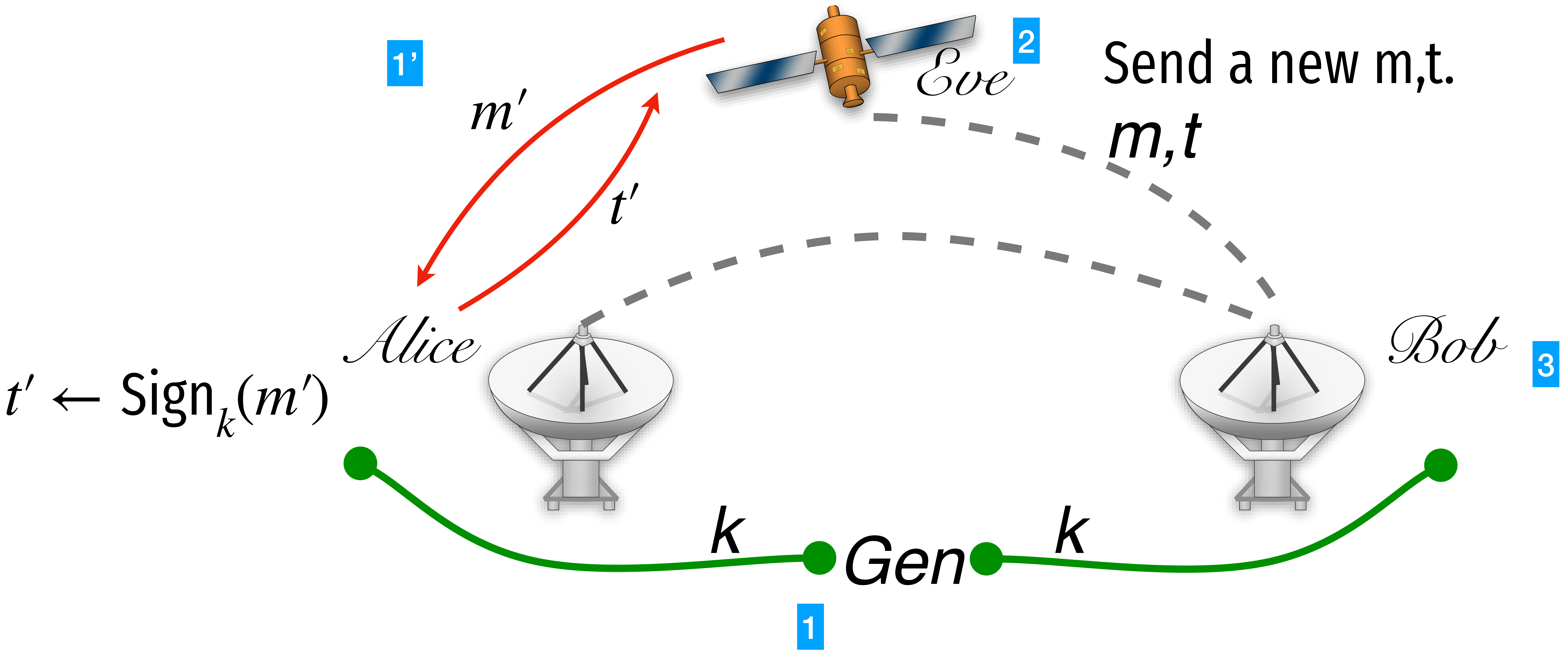


Security for a MAC (similar to Signature)

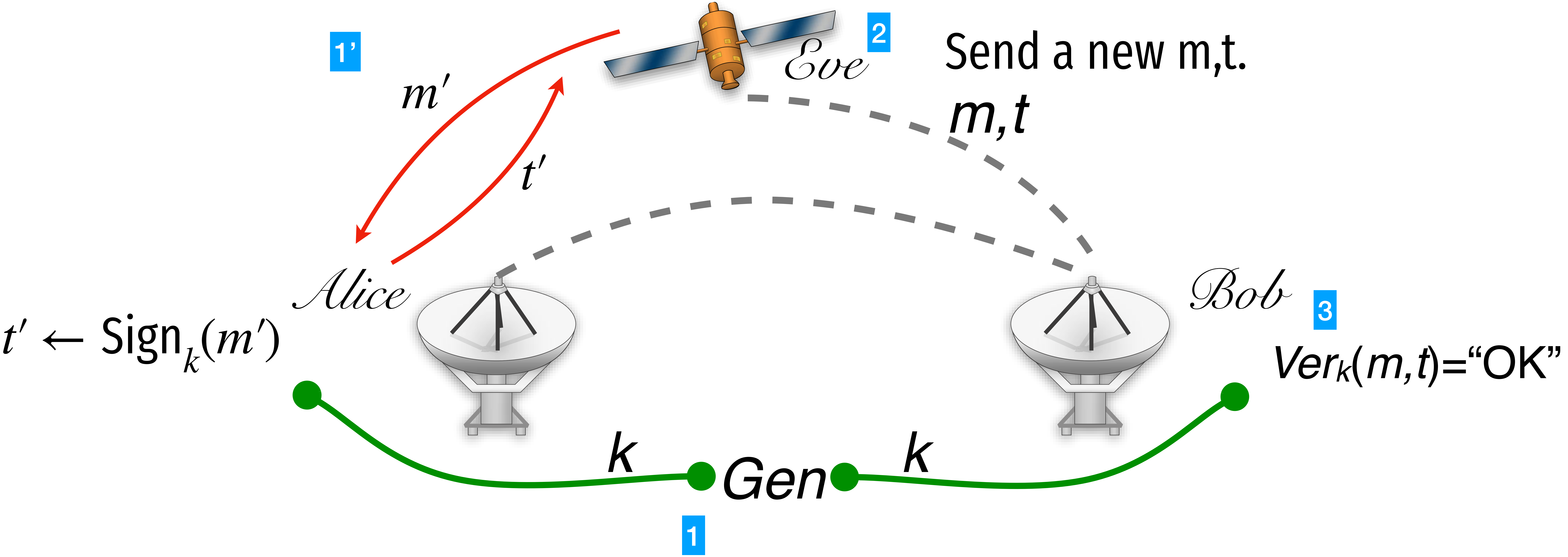
$t' \leftarrow \mathsf{Sign}_k(m')$



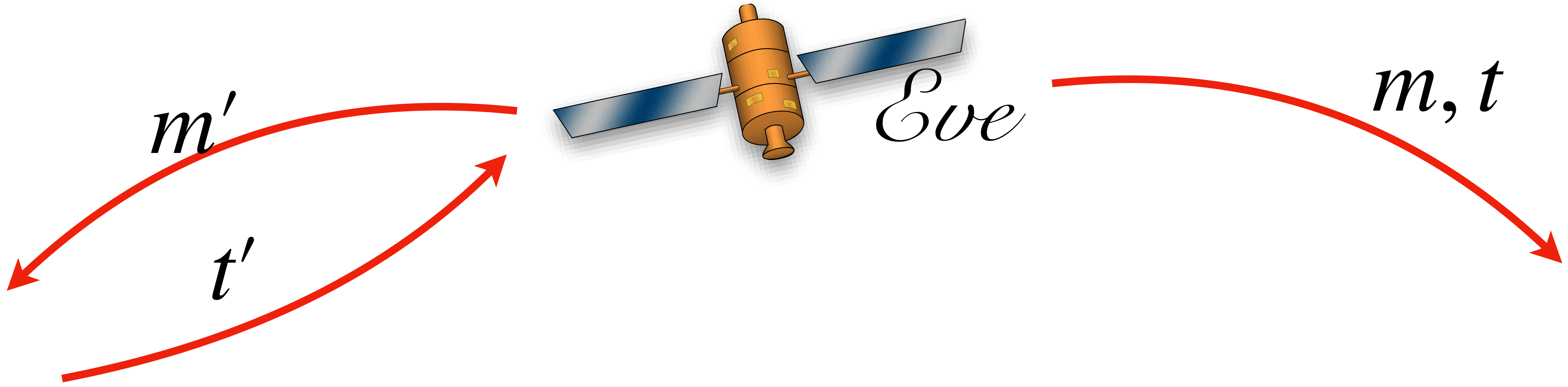
Security for a MAC (similar to Signature)



Security for a MAC (similar to Signature)



Security intuition



$$\Pr[F_k(m) = t] =$$

Lets do some class exercises with these tools.