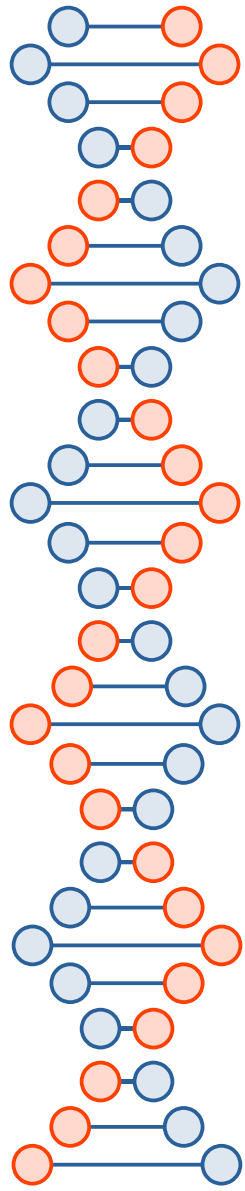


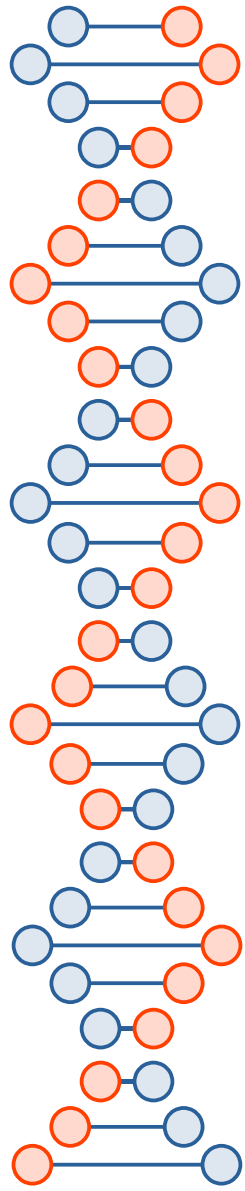
RSA Semantic Security

CSE 468 Fall 2026
jedimaestro@asu.edu

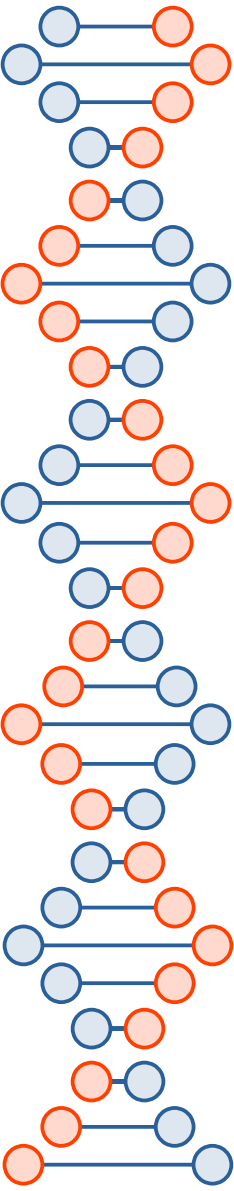


$$n = pq$$

- p and q are primes, suppose $p = 61$, $q = 53$
- $n = 3233$
- Euler's totient counts the positive integers up to n that are relatively prime to n
 - $(61 - 1)(53 - 1) = 3120$
- Carmichael's totient function = $\text{lcm}(60, 52)$
 - 52, 104, 156, 208, 260, 312, 364, 416, 468, 520, 572, 624, 676, 728, 780
 - 60, 120, 180, 240, 300, 360, 420, 480, 540, 600, 660, 720, 780
- Choose $1 < e < 780$ coprime to 780, e.g., $e = 17$
- d is the modular multiplicative inverse of e , $d = 413$
- $413 * 17 \bmod 780 = 1$



- Public key is $(n = 3233, e = 17)$
- Private key is $(n = 3233, d = 413)$
- Encryption: $c(m = 65) = 65^{17} \bmod 3233 = 2790$
 - What if m is a multiple of 61 or 53?
- Decryption: $m = 2790^{413} \bmod 3233 = 65$
- But, signatures are a more important application of RSA...
 - Signature: $s = 100^{413} \bmod 3233 = 1391$
 - Verification: $100 = 1391^{17} \bmod 3233$
- Fast modular exponentiation is the trick
 - Also need extended Euclidean algorithm
- Using RSA for key exchange or encryption is often a red flag, more commonly used for signatures

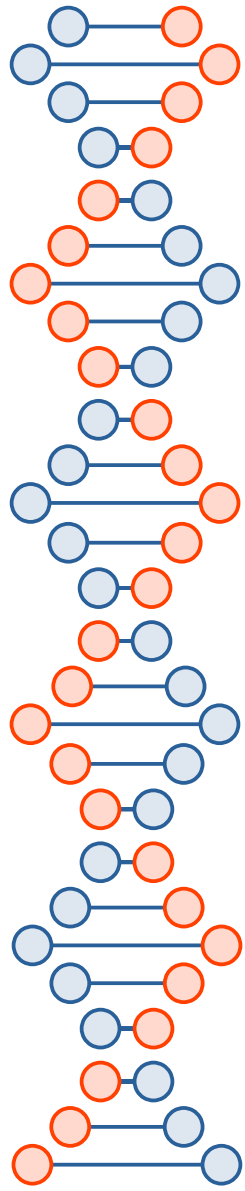


RSA in real cryptosystems

- What we learned, and what you read about in the original RSA paper, is called “Textbook RSA”
 - Not secure and should not be used (padding is strictly necessary in real schemes)
 - Padding oracle attacks (same idea as for CBC)
 - A type of side channel

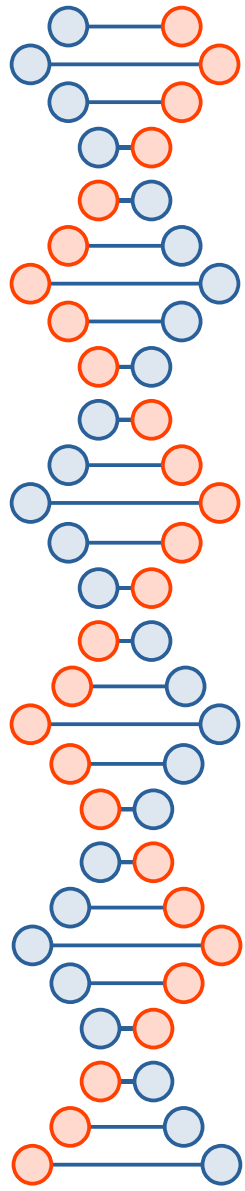
Symmetric attack types

- Ciphertext only
 - Think Caesar cipher, or Viginere cipher
- Known plaintext
 - Linear cryptanalysis
- Chosen plaintext
 - Differential cryptanalysis



Asymmetric notions of semantic security

- Now threat models are very complicated, but in a nutshell:
 - IND-CPA – Indistinguishability under chosen plaintext attack
 - IND-CCA – Indistinguishability under chosen ciphertext attack
 - IND-CCA2 – Indistinguishability under chose ciphertext attack (adaptive)



A preliminary version of this paper appeared in *Advances in Cryptology – Eurocrypt 94 Proceedings*, Lecture Notes in Computer Science Vol. 950, A. De Santis ed., Springer-Verlag, 1994.

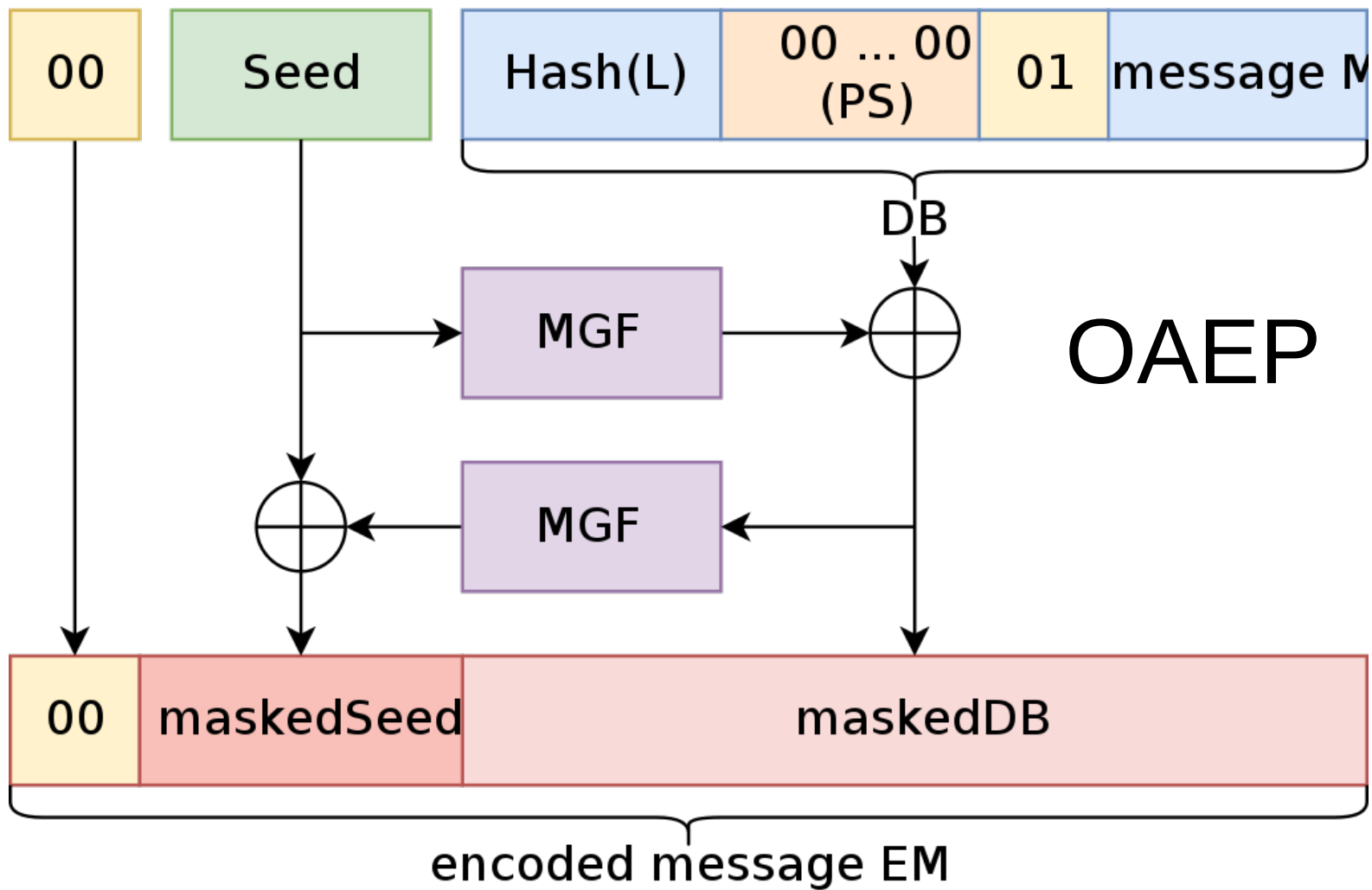
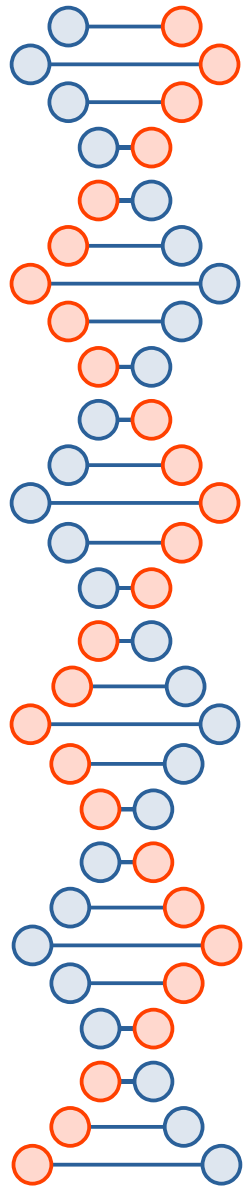
Optimal Asymmetric Encryption — How to Encrypt with RSA

MIHIR BELLARE*

PHILLIP ROGAWAY†

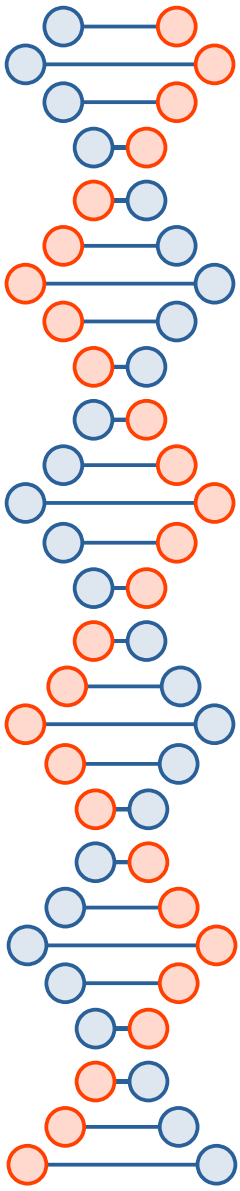
November 19, 1995

<https://cseweb.ucsd.edu/~mihir/papers/oaep.pdf>



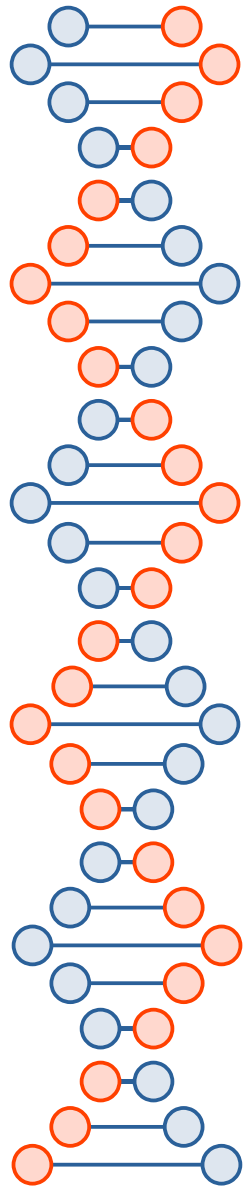
IND-CCA2 in a nutshell

- I'll encrypt or decrypt as many plaintexts or ciphertexts as you like
 - plaintext/ciphertext pairs
- You give me two plaintexts, I'll flip a coin (heads or tails) and encrypt one of them (you don't know which) to give you C
- In polynomial time, you can do more encryption and decryption, just not for C
- You guess my coin flip (heads or tails)



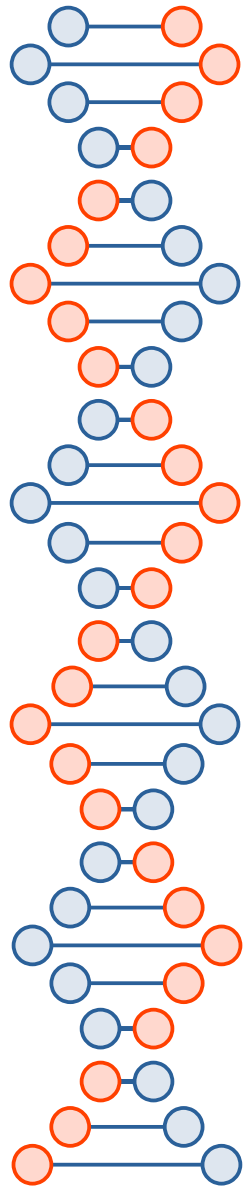
If you can't win with $>50\%$ probability

- You can't break my scheme (e.g., OAEP) with an adaptive chosen ciphertext attack

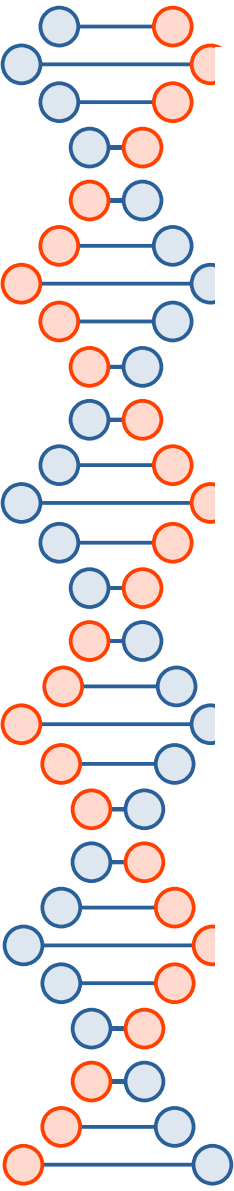


If you **can** win with $>50\%$ probability

- You've potentially broken my scheme with an adaptive chosen ciphertext attack
- Let's win the Turing award together, by publishing a paper showing how to factor large integers with a classical computer in polynomial time
 - Or, build a cybercrime cartel together?



Okay to grab the RSA paper and start coding?
Or just use a textbook, *i.e.*, textbook RSA?



Let C be the RSA encryption of 128-bit AES key k with RSA public key (n, e) . Thus, we have

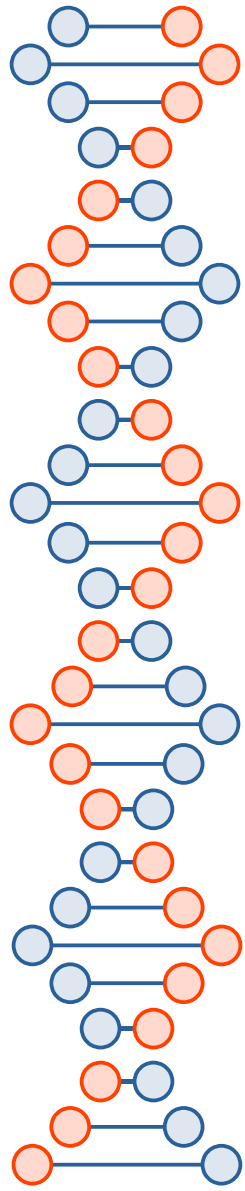
$$C \equiv k^e \pmod{n}$$

Now let C_b be the RSA encryption of the AES key

$$k_b = 2^b k$$

i.e., k bitshifted to the left by b bits. Thus, we have

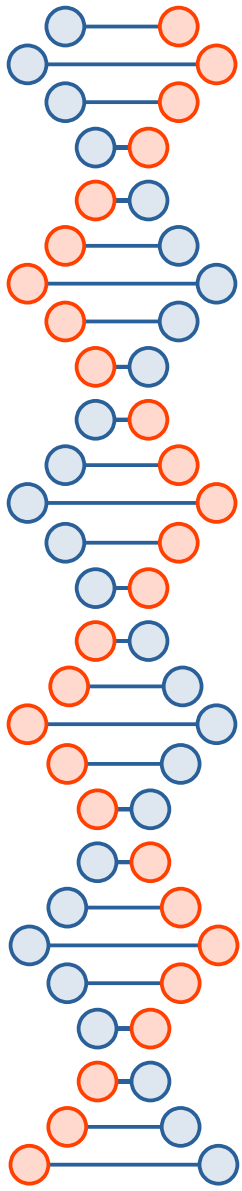
$$C_b \equiv k_b^e \pmod{n}$$



$$C_b \equiv k_b^e \pmod{n}$$

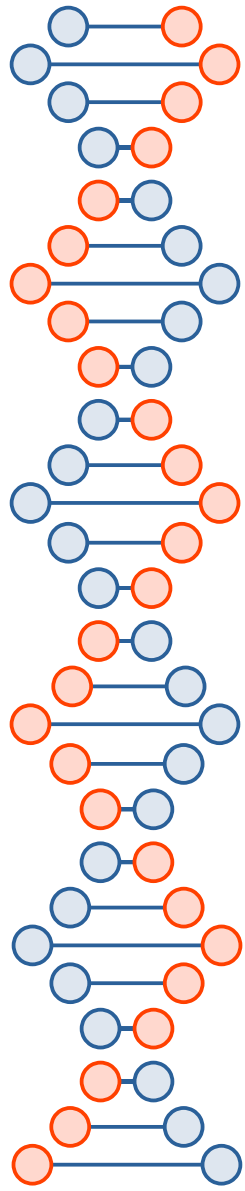
We can compute C_b from only C and the public key, as

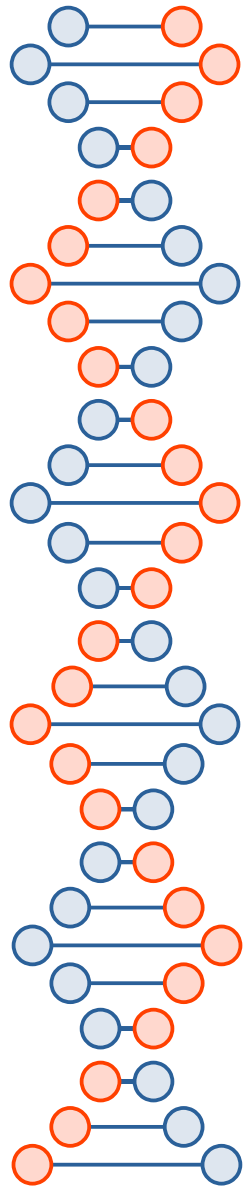
$$\begin{aligned} C_b &\equiv C(2^{be} \bmod n) \pmod{n} \\ &\equiv (k^e \bmod n)(2^{be} \bmod n) \pmod{n} \\ &\equiv k^e 2^{be} \pmod{n} \\ &\equiv (2^b k)^e \pmod{n} \\ &\equiv k_b^e \pmod{n} \end{aligned}$$



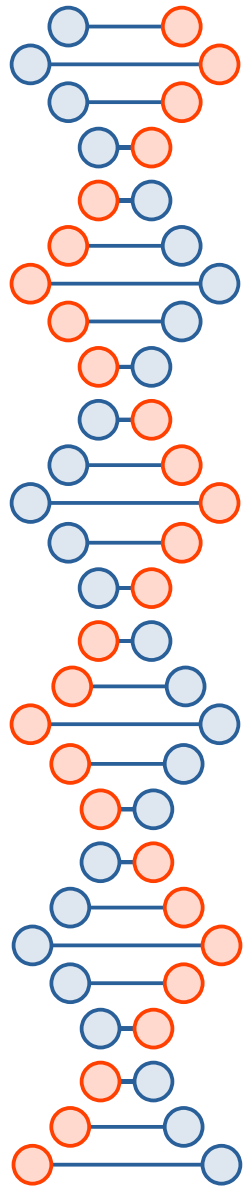
Suppose the client whose communications we want to decrypt encrypts the following 128-bit AES key with 1024-bit RSA and sends it to the server:

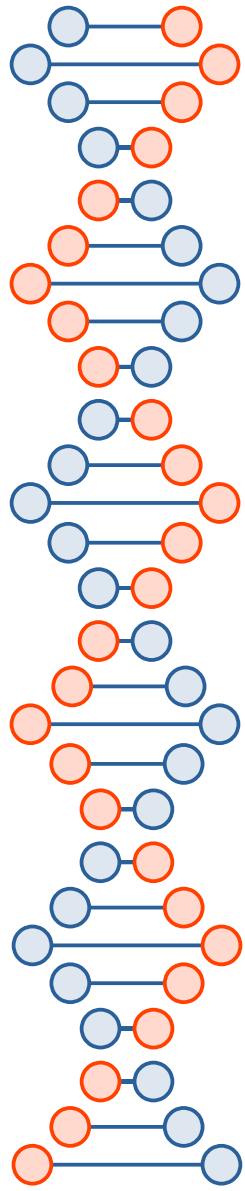
```
1011000010010110011101111011101100100010111111001110101011110011  
0000000011100100101111001001010100100011101101010000101110111011
```

[illegible]

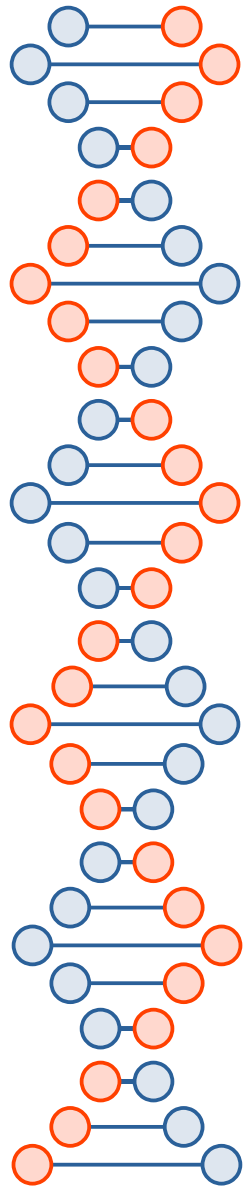


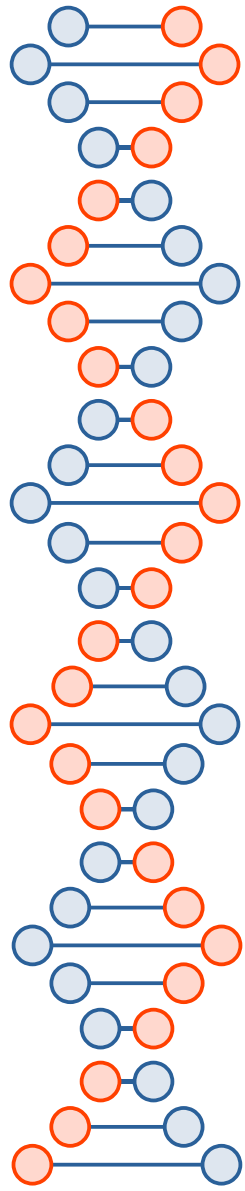
This shows what the server sees as the RSA plaintext *after* it decrypts the ciphertext we sent it, which we are trying to trick the server into leaking bits of to us. So, we as the attacker have recorded c by eavesdropping on the client and server's communications over the Internet. But without the private key, d , we don't know what m (the green part, which is the AES key to decrypt the rest of the message) is. Let's explore what happens if we open our own connection to the server, and as our ciphertext we send $c \times 2^e$. The server will decrypt that into the following plaintext:

[illegible]

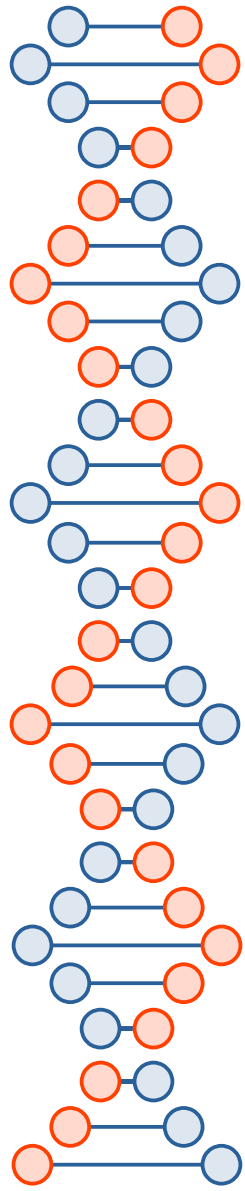


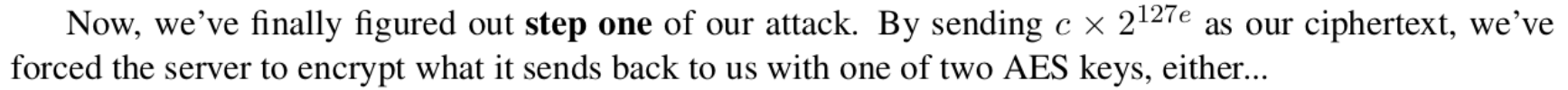
So, we know how to double plaintexts by manipulating ciphertexts. What if we double it more than once? What if we do it 16 times, by sending $c \times 2^{16e}$ as our ciphertext. When we multiplied the plaintext by 2 above, we effectively bit shifted the AES key by 1. Now we're multiplying the plaintext by 2^{16} , which is the equivalent of bit-shifting to the left 16 times:

[illegible]



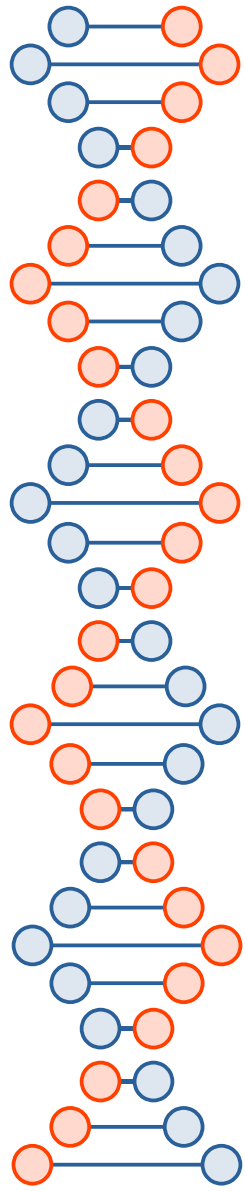
$$c \times 2^{127e}$$

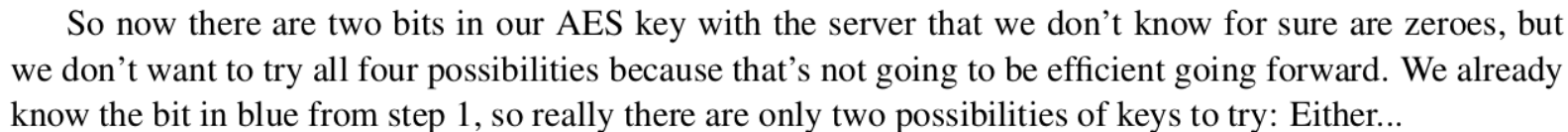
[illegible]



...Or...

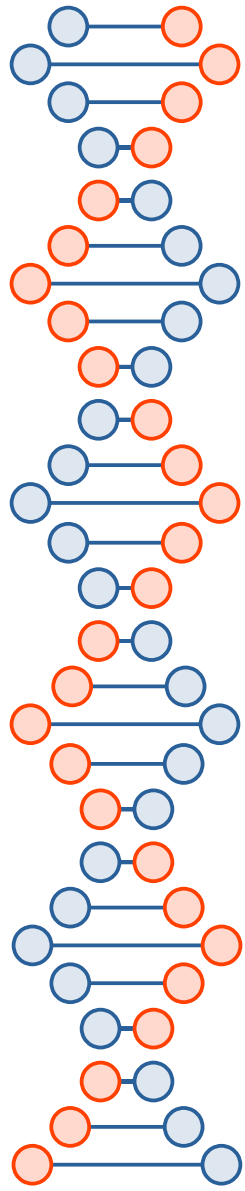
23

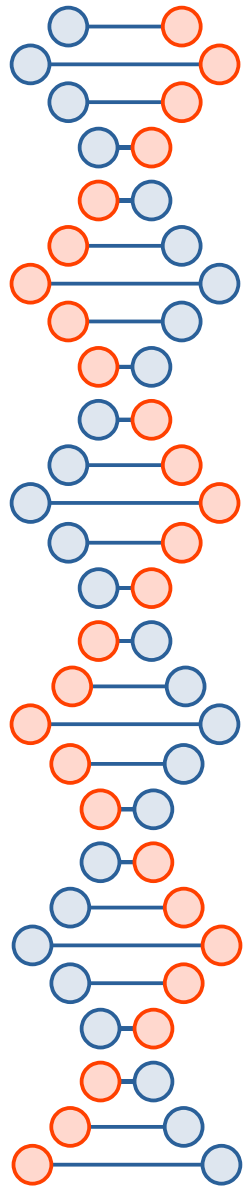
[illegible]



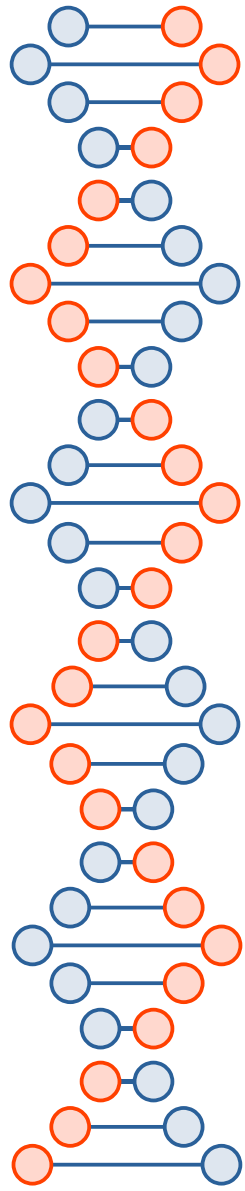
...Or...

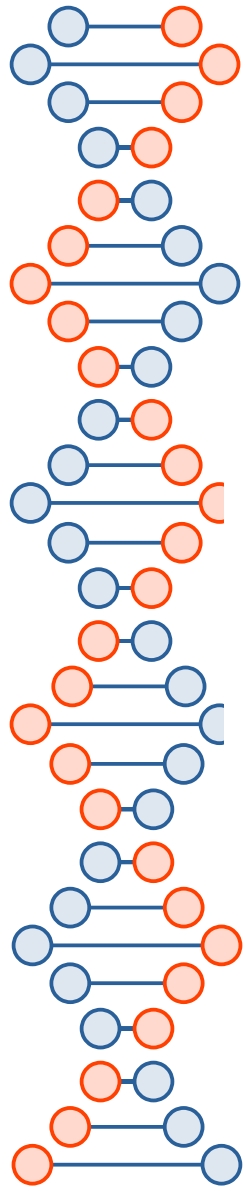
So then, for step 3, we would send $c \times 2^{125e}$ as our ciphertext, and the server would decrypt:

[illegible]



By the 128th step...

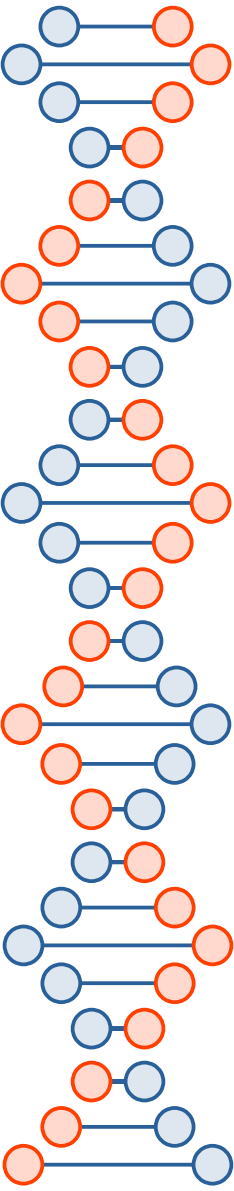
[illegible]



...Or...

```
0011000010010110011101111011101100100010111111001110101011110011
0000000011100100101111001001010100100011101101010000101110111011
```

```
1011000010010110011101111011101100100010111111001110101011110011
0000000011100100101111001001010100100011101101010000101110111011
```



WUP requests, textbook RSA

- Full attack is at: <https://arxiv.org/pdf/1802.03367.pdf>
- Required reading for Thursday:
<https://m0na.net/papers/wirewatch.pdf>
- See also:
<https://luca-giuzzi.unibs.it/corsi/Support/papers-cryptography/10.1.1.37.9316.pdf>

Takeaways

- Generally, DH for key exchange and RSA for signatures
 - Alternatives, such as elliptic curves
- Symmetric crypto for the actual encryption
- Semantic security is the gold standard for asymmetric
 - Reduction proofs
- “Textbook RSA,” like you’ll do on the exam, is dangerous