Solutions to Daily Due Sep 16 (AI generated)

Math 4440/5440, Coding and Cryptography, Fall 2026

Generated by Claude for Katherine E. Stange
(all computations computer-verified)

Problem 1 was review for Test 1, with nothing to hand in. What follows is a complete worked pass through the optional El Gamal encryption chain (Problem 2), including the two "challenges for fun" at the end.

The parameters

As on the Diffie-Hellman Tools page,

$$p = 10^{15} + 37 = 1000000000000037, \qquad g = 2.$$

This p is prime, and 2 is in fact a primitive root modulo p (its order is exactly $p-1$; this was checked). Its factorization will matter at the end:

$$p - 1 = 2^2 \cdot 7 \cdot 37 \cdot 965250965251.$$

(1)–(2) Keys and the message

Bob's key pair. He picks a private b and publishes $B = g^b$:

$$b = 314159265358979, \qquad B = 2^b \bmod p = 427277223625969.$$

(The private key is just a number he keeps; the public key is what goes on `#public-keys`.)

Alice's message. Answering "Best food?" with `tacos`, the *Text to Integer* tool gives, reading the letters as digits base 255,

$$m = 116 + 97 \cdot 255 + 99 \cdot 255^2 + 111 \cdot 255^3 + 115 \cdot 255^4 = 488095816826,$$

comfortably less than p.

(3) Encrypting to Bob

Alice chooses a fresh *ephemeral* exponent k — new for every message — and computes

$$k = 271828182845904, \qquad K = g^k \bmod p = 42422796446951,$$

$$c = m \cdot B^k \bmod p = 591098269650176.$$

She posts the pair

$$(K, c) = (42422796446951,\ 591098269650176)$$

on `#ciphertexts`, tagging Bob.

(4)–(5) Bob decrypts

Bob computes the shared secret from his private key,

$$s = K^b \bmod p = 480911367614278,$$

which is the same number as B^k that Alice used, because both equal g^{bk}. Then

$$m = c \cdot s^{-1} \bmod p = 488095816826,$$

and Integer to Text turns this back into `tacos`. In Sage the inverse is `inverse_mod(s, p)` (or use $s^{p-1-b\cdot k}$-style tricks; the inverse is easier).

A useful shortcut some people used instead: rather than inverting, compute s^{-1} directly as $K^{p-1-b} \bmod p$, since $K^{p-1} \equiv 1$.

Challenge (i): break someone else's message

Breaking a ciphertext means solving the discrete logarithm problem: find b from $B = g^b$ (or k from K), then decrypt as Bob does. With this p it is genuinely doable, because $p-1$ is *smooth enough*:

$$p - 1 = 2^2 \cdot 7 \cdot 37 \cdot 965250965251.$$

Pohlig-Hellman reduces the discrete log modulo p to discrete logs in each prime-power factor: trivial work modulo 4, 7 and 37, and then one hard piece of size $965250965251 \approx 9.7 \times 10^{11}$. **Baby-step giant-step** on that piece needs about

$$\sqrt{965250965251} \approx 982472$$

baby steps — a table of about a million entries, which is a few seconds of computing and a few tens of megabytes. So the whole key is recoverable on a laptop.

That is the point of the exercise: 10^{15} looks like a big number, but $\sqrt{p} \approx 3 \times 10^7$ operations is nothing, and even the Pohlig-Hellman-reduced cost of 10^6 is less. Real Diffie-Hellman and El Gamal use p of 2048 bits or more *and* insist that $p-1$ have a large prime factor, precisely to block both attacks.

Challenge (ii): the repeated ephemeral key

Suppose Alice is careless and uses the *same* k for two messages m_1 and m_2 to the same person. Then

$$c_1 = m_1 B^k, \qquad c_2 = m_2 B^k \pmod p,$$

and the unknown B^k cancels in the ratio:

$$\frac{c_2}{c_1} \equiv \frac{m_2}{m_1} \pmod p.$$

An eavesdropper who knows (or guesses) one plaintext gets the other immediately. With the numbers above, the second message `pizza` is $m_2 = 412171198312$ and $c_2 = 432377801555909$; then

$$c_2 c_1^{-1} \bmod p = 733006565117842 = m_2 m_1^{-1} \bmod p,$$

and multiplying by the known $m_1 = 488095816826$ returns $412171198312 =$ `pizza`. Even with neither plaintext known, the ratio m_2/m_1 leaks a lot: for short English words it is enough to search.

The same failure destroyed the PlayStation 3 code signing keys in 2010, where a repeated nonce in ECDSA revealed Sony's private key. **Never reuse an ephemeral key.**