CPSC 418 / MATH 318 — Introduction to Cryptography
ASSIGNMENT 3

Name: Cody Clark
Student ID: 30010560

Problem 1 — Flawed MAC designs, 13 marks

(a) ***Proof.*** Let M_1 be a message consisting of L blocks $P_1, P_2, \ldots, P_L$, each of length n such that $M_1 = P_1||P_2||\ldots||P_L$. Let M_2 be a message where $M_2 = M_1||X$, where X is an arbitrary n-bit block. We now define a message authentication function $PHMAC_K(M) = ITHASH(K||M) = ITHASH(K||P_1||P_2||...||P_L)$, where K is an n-bit key and $ITHASH$ is the hash function defined in the assignment. Suppose an attacker knows a message/PHMAC pair $(M_1, PHMACK(M_1))$.

We will now show that an attacker can compute $PHMAC_K(M_2)$ without knowledge of K. To begin we will examine the steps taken in order to compute $PHMACK(M_1)$ by the $ITHASH$ function. (For this example we assume $L > 6$ in order to show the pattern, but by removing some steps L could be as small as 1)

After initializing $H = 0^n$ we compute the following for the first round $(i = 1) : H_1 = f(H^0, P_1)$ which is equivalent to $H_1 = f(0^n, K)$. For the subsequent rounds we can see the following pattern emerge:

$$\begin{aligned}
(i = 2) : H_2 &= f(H_1, P_2) \\
&= f(f(H_0, P_1), P_2) \\
&= f(f(0^n, K), P_2) \\
(i = 3) : H_3 &= f(H_{2,P}\, 3) \\
&= f(f(f(0^n, K), P_2), P_3) \\
&\vdots \\
(i = L) : H_L &= f(H_{L-1}, P_L) \\
&= f(f(...f(f(f(0^n, K), P_2), P_3), \ldots), P_{L-1}), P_L) = PHMAC_K(M_1)
\end{aligned}$$

The process to compute$PHMAC_K(M_2)$ if the key K were known would be as follows:

$$\begin{aligned}
PHMAC_K(M_2) &= ITHASH(K||M_2) \\
&= ITHASH(K||M_1||X) \\
&= ITHASH(K||P_1||P_2||...||P_L||X)
\end{aligned}$$

If the key K were known, then the $ITHASH$ algorithm would proceed as follows:

$$\begin{aligned}
(i=1): H_1 &= f(H^0, P_1) \\
&= f(0^n, K) \\
(i=2): H_2 &= f(H_1, P_2) \\
&= f(f(H_0, P_1), P_2) \\
&= f(f(0^n, K), P_2) \\
(i=3): H_3 &= f(H_2, P_3) \\
&= f(f(f(0^n, K), P_2), P_3) \\
&\vdots \\
(i=L): H_L &= f(H_{L-1}, P_L) \\
&= f(f(...f(f(f(0^n, K), P_2), P_3), \dots), P_{L-1}), P_L)
\end{aligned}$$

(Finally we compute the hash $f(H_L, X)$)

$$\begin{aligned}
(i=L+1): H_X &= f(H_L, X) \\
&= f(f(f(...f(f(f(0^n, K), P_2), P_3), \dots), P_{L-1}), P_L), X) \\
&= PHMAC_K(M_2)
\end{aligned}$$

Knowing $PHMAC_K(M_1) = H_L$ however, we can use that as a substitution as follows:

$$(i=L+1): H_X = f(PHMAC_K(M_1), X) = PHMAC_K(M_2)$$

This requires no knowledge of the key K, just knowledge of $PHMAC_K(M_1)$. Thus we have seen how an attacker could produce $PHMAC_K(M_2)$. QED.

(b) ***Proof.***(In order to render this answer shorter, assume all values are as defined in the assignment)

We will now show how an attacker can find a second message/AHMAC pair.

We know that $ITHASH$ is not weakly collision resistant. That is, we know it is computationally feasible that, given M_1, an attacker can find an M_2 such that $ITHASH(M_1) = ITHASH(M_2)$. We assume that such an M_2 has been found. (In order to find M_2 one might exploit some property of the public function f, but in this example we don't know the details of f.) Let us examine the following steps of $ITHASH$ used to calculate $ITHASH(M_1)$ as we did in a).

$$
\begin{aligned}
(i=1): H_1 &= f(H_0, P_1) \\
&= f(0^n, P_1) \\
(i=2): H_2 &= f(H_1, P_2) \\
&= f(f(H_0, P_1), P_2) \\
&= f(f(0^n, P_1), P_2) \\
&\vdots \\
(i=L): H_L &= f(H_{L-1}, P_L) \\
&= f(f(...f(f(0^n, P_1), P_2), \dots), P_{L-1}), P_L) = ITHASH(M_1)
\end{aligned}
$$

In order to calculate $AHMAC_K(M_1)$ an additional last step would need to be performed:

$$
\begin{aligned}
(i=L+1): H_L &= f(f(f(...f(f(0^n, P_1), P_2), \dots), P_{L-1}), P_L), K) = AHMAC_K(M_1) \\
&= f(ITHASH(M_1), K) = AHMAC_K(M_1)
\end{aligned}
$$

Because we know it is computationally feasible to find a message M_2 such that $ITHASH(M_1) = ITHASH(M_2)$, and we have assumed such a message M_2 has been found, we can substitute the last statement to the following:

$$
(i=L+1): H_L = f(ITHASH(M_2), K) = AHMAC_K(M_1) = AHMACK(M_2)
$$

This is possible because we know the compression function f is public knowledge, therefore it is easy to compute $ITHASH(M_1)$ and $ITHASH(M_2)$ (Neither of which require the key K), and the first L rounds of computing $AHMAC_K(M)$ do not depend on K. In essence, an attacker who knows $M_1, AHMAC_K(M_1)$ and can computationally find an M_2 where $ITHASH(M_1) = ITHASH(M_2)$, has found a new pair $(M_2, AHMAC_K(M_2))$ where they know $AHMAC_K(M_2) = AHMAC_K(M_1)$. QED.

Problem 2 — Fast RSA decryption using Chinese remaindering, 8 marks)

Proof. In order to prove this method correctly works we must prove that $M = M$'. That is, we must prove:

$$C^d = M = M' \equiv pxM_q + qyM_p \ (mod\ n)$$

where $0 \leq M \leq n - 1$.

To prove this we will use the chinese remainder theorem. This theorem states that given pairwise coprime positive integers $n_1, n_2, \ldots, n_k$ and arbitrary integers $a_1, a_2, \ldots, a_k$, the system of simultaneous congruences

$$\begin{aligned} X &\equiv a_1 \ (mod\ n_1) \\ X &\equiv a_2 \ (mod\ n_2) \\ &\vdots \\ X &\equiv a_k \ (mod\ n_k) \end{aligned}$$

has a solution, and the solution is unique modulo $N = n_1 n_2 ... n_k$.

An ordinary RSA scheme to decrypt a message we would have to calculate $M \equiv C^d \ (mod\ n)$. Using the Chinese remaindering theorem we can see that because p and q are coprime if:

$$M \equiv C^d \ (mod\ n) \equiv C^d \ (mod\ pq)$$

then

$$\begin{aligned} M &\equiv C^d \ (mod\ p) \\ M &\equiv C^d \ (mod\ q) \end{aligned}$$

By solving for M by the CRT Alice can decrypt C. Let's move through this process step by step. Throughout this proof I will refer to $(p-1)$ and $(q-1)$ as $\phi(p)$ and $\phi(q)$ respectively because we know that p and q are prime. For step 1 and 2 the procedure is applied identically using q instead of p, so showing the procedure using p is sufficient.

(Step 1): In calculating $d_p \equiv d \ (mod\ \phi(p))$ we are essentially factoring d. $d = k\phi(p) + d_p$ for some integer k. As we will see in step 2, d_p is the only truly relevant number.

(Step 2): Because we know that both p and q are prime, we can use this factorization to apply Euler's theorem:

$$\begin{aligned} M &\equiv C^d \ (mod\ p) \\ &= C^{\phi(p)*k+d_p} \ (mod\ p) \qquad \text{- By step 1} \\ &= (C^{\phi(p))k} C^{d_p} \ (mod\ p) \\ &= (1)kC^{d_p} \ (mod\ p) \qquad \text{- By Euler's Theorem} \\ &= C^{d_p} \equiv M_p \ (mod\ p) \end{aligned}$$

Where $1 \leq M_p \leq \phi(p)$, which is a much more manageable number than C^d. We have thus far shown that $M \equiv M_p \equiv C^d \ (modp)$ and $M \equiv M_q \equiv C^d \ (mod\ q)$. The steps taken haven't really gotten Alice closer to decrypting the message, rather they've enabled decryption using more manageable numbers. At this point Alice must then compute an M which satisfies both of these congruences. That is:

$$M \equiv M_p \ (mod\ p)$$
$$M \equiv M_q \ (mod\ q)$$

(Step 3): We know a few things at this point: $M = px + M_p$ (Because $M \equiv M_p \ (mod\ p)$ implies px divides M with remainder of M_p for some x) and $M = qy + M_q$. (Because $M \equiv M_q \ (mod\ q)$ implies qy divides M with remainder of M_q for some y) Therefore we need to solve the following equations which we know have solutions because both p and q are prime:

Case 1: When the Euclidean Algorithm takes an even number of steps to complete
$$px + q(-y) = 1 \rightarrow px \equiv 1 \ (mod\ q) \ (x \text{ is } p\text{'s inverse for } (mod\ q))$$
Case 2: When the Euclidean Algorithm takes an odd number of steps to complete
$$p(-x) + qy = 1 \rightarrow qy \equiv 1 \ (mod\ p) \ (y \text{ is } q\text{'s inverse for } (mod\ p))$$

These cases are generalized as step 3: $px+qy = 1$. This equation is solved using the Extended Euclidean Algorithm.

(Step 4): Now we will use x and y from step 3 in order to calculate the solution to our series of congruences.

We will explore $M \equiv pxM_q + qyM_p \ (mod\ pq)$.

Case 1:

$M \equiv pxM_q + qyM_p \ (mod\ p)$
$\equiv pxM_q + (1)M_p \ (mod\ p)$ - Because $qy \equiv 1 \ (mod\ p)$, y being the inverse of q
$\equiv (0) + M_p \ (mod\ p)$ - Because p divides pxM_q without remainder
$\equiv M_p \ (mod\ p)$ - Therefore M_p is congruent to $M \ (mod\ p)$

Case 2:

$M \equiv pxM_q + qyM_p \ (mod\ q)$
$\equiv (1)M_q + qyM_p \ (mod\ q)$ - Because $px \equiv 1 \ (mod\ q)$, x being the inverse of p
$\equiv M_q + (0) \ (mod\ q)$ - Because q divides qyM_p without remainder
$\equiv M_q \ (mod\ q)$ - Therefore M_q is congruent to $M \ (mod\ q)$

Therefore we have found a solution to the system of congruences which satisfies both of the equations we initially needed to satisfy, and then by the CRT $M \equiv C^d \ (mod\ n)$ using the process described. We will now examine the vanilla RSA method to ensure that $M \equiv C^d \ (mod\ n)$ as we assumed at the start. We can prove that $C^d \equiv M \ (mod\ n)$ through the method described within the course slides. $C^d \equiv (M^e)^d \equiv M^{ed} \ (mod\ n)$, with d chosen such

that it's the inverse $ed \equiv 1 \ (mod \ \phi(n))$ and then by applying Euler's theorem to M^{ed} we can arrive at the result of $C^d \equiv M \ (mod \ n)$. (As shown in the course lides by Renate Scheidler)

Finally we can conclude that

$$C^d = M = M' \equiv pxM_q + qyM_p \ (mod \ n)$$

where $0 \leq M \leq n - 1$. QED.

Problem 3 — RSA primes too close together, 18 marks)

(a) ***Proof.*** Let n, p, and q be integers defined as they are in a typical RSA cryptosystem, that is, $n = pq$, where p and q are primes. Suppose n can be represented as a difference of squares such that $n = x^2 - y^2$, where $0 < y < x < n$, and also suppose in this instance that $p > q$, and all roots are positive.

x is defined as an integer such that $x = \frac{(p+q)}{2}$, which is the average of p and q. Therefore p or q must be larger than x, with the other being smaller. We have defined $p > q$, therefore $p > x$.

Suppose there exists two pairwise integers, a, b, such that $a = b$, and $ab = n$. ($a = b = \sqrt{n}$). Suppose there exists an integer c such that $c = a+1$. Consider what happens when we replace a by c: $cb = n + b$. In order for n to be factorized there must be an integer d such that $d < a$, $b < c$, so that $cd = n$. Therefore, for any p, q where $pq = n$ and $p > q$, p must be at least as large as c and q must be at least as small as d. $p \geq c > a$, b, $\sqrt{n} > d \geq q$. Therefore $p > \sqrt{n}$.

At this point we have shown that $p > x$, $\sqrt{n}$. Now we must show that $x > \sqrt{n}$. n is defined such that $n = x^2 - y^2$. Suppose $\sqrt{n} \geq x$. Then $n \geq x^2$ such that $n = x^2 + k$, where k is some non-negative integer. However by n's definition $n + y^2 = x^2$, so that a positive number must be added to n in order to equal x^2, proving by contradiction that $x > \sqrt{n}$.

Thus $y > x > \sqrt{n}$.

QED

(b) ***Proof.***

(while clause): When $a = x$ then $b = \sqrt{x^2 - n} = \sqrt{y^2} = y$, an integer.

(no early termination): Suppose the algorithm terminated when $a < x$. This implies that there exists a value $a < x$ such that $b = \sqrt{a^2 - n}$ where b is an integer. Then:

$$b^2 = a^2 - nn \qquad\qquad = a^2 - b^2 n = (a + b)(a - b)$$

We know that the prime factorization of n is the trivial case $n = n \cdot 1$, and the case $n = pq$, $p > q$. Therefore $p = a + b$ and $q = a - b$. We know that $p = x + y$, and $q = x - y$. If $a < x$, then there exists a value i such that $x = a + i$. Then there would need to be a solution to the system of equations $p = x - i + y$ and $q = x - i - y$, where $n = pq$. Because the only ways to factor n is by $n = n \cdot 1$ and $n = pq = (x + y)(x - y)$, and there's no solution to the system of equations, there exists a contradiction proving that a must be equal to x and will not terminate for any values less than x.

(output = q): Suppose $a = x$. Then $b = \sqrt{x^2 - n}$, $b^2 = x^2 - n = y^2$, therefore $b = y$. Then $a - b = x - y = q$.

QED

(c) ***Proof.*** a begins at $\lceil\sqrt{n}\rceil$. The goal is for the algorithm to terminate when $a = x$, with each iteration increasing a by 1. Therefore one only needs to count the number of integers between $\lceil\sqrt{n}\rceil$ and x. This can be calculated by $x - \lceil\sqrt{n}\rceil$. That is, by the end of the algorithm $a = \lceil\sqrt{n}\rceil + i = x$ for some integer i. In addition the algorithm must at least set up the values of a and b which is always counted as 1 iteration and provides the lower bounds when $x - \lceil\sqrt{n}\rceil = 0$ because $x = \lceil\sqrt{n}\rceil$.

(d) ***Proof.*** We see that $y^2 = x^2 - n$, therefore:

$$\begin{aligned} x - \lceil\sqrt{n}\rceil &< \frac{x^2 - n}{2\sqrt{n}} \\ = x - \lceil\sqrt{n}\rceil &< \frac{(x - \sqrt{n})(x + \sqrt{n})}{2\sqrt{n}} \qquad \text{- By the binomial theorem} \\ = (x - \lceil\sqrt{n}\rceil)(2\sqrt{n}) &< (x - \sqrt{n})(x + \sqrt{n}) \end{aligned}$$

Now we can consider each pair of terms individually. We can see that $x - \lceil\sqrt{n}\rceil$ must be equal or smaller than $x - \sqrt{n}$, we have also proved through part a) that $x > \sqrt{n}$, so $(2\sqrt{n}) = (\sqrt{n} + \sqrt{n}) < (x + \sqrt{n})$. Therefore the inequality holds.
QED.

(e) ***Proof.*** We know from c) that the number loop iterations executed by the algorithm is $x - \lceil\sqrt{n}\rceil$. Therefore we must prove:

$$\begin{aligned} x - \lceil\sqrt{n}\rceil + 1 &\leq \frac{B^2}{2} + 1 \\ = x - \lceil\sqrt{n}\rceil &\leq \frac{B^2}{2} \\ = 2(x - \lceil\sqrt{n}\rceil) &\leq B^2 \end{aligned}$$

We also know that $p - q = y < 2B\sqrt[4]{n}$. Thus:

$$\begin{aligned} \frac{y}{2} &< B\sqrt[4]{n} \\ = \frac{y}{2\sqrt[4]{n}} &< B \\ = \frac{y^2}{2\sqrt[4]{n}} &< B^2 \end{aligned}$$

We've proved in d) that $x - \lceil\sqrt{n}\rceil < \frac{y^2}{2\sqrt{n}}$. Thus:

$$2(x - \lceil\sqrt{n}\rceil) < \frac{y^2}{\sqrt{n}}$$

Finally we can prove $\frac{y^2}{\sqrt{n}} \leq \frac{y^2}{2\sqrt[4]{n}}$. Suppose $n = ss$ for some integer $s \geq 2$. Suppose $s = tt$ for some integer $t \geq 2$. (Otherwise $s = 1$ and we know $n > 1$) Then we can see:

$$\frac{y^2}{\sqrt{n}} = \frac{y^2}{s} \leq \frac{y^2}{2\sqrt[4]{n}} = \frac{y^2}{2t}$$

Therefore we can conclude:

$$2(x - \lceil\sqrt{n}\rceil) < \frac{y^2}{\sqrt{n}} \leq \frac{y^2}{2\sqrt[4]{n}} < B^2 \qquad \text{where} \qquad x - \lceil\sqrt{n}\rceil < \frac{B^2}{2}$$

which shows that the algorithm actually factors n with strictly less than $\frac{B^2}{2} + 1$ loop iterations. QED.

Problem 4 – The El Gamal public key cryptosystem is not semantically secure, 12 marks

Proof. Let M_1 and M_2 be plaintexts where $M_1 \in QR_p$ and $M_2 \in NR_p$. Let C be a ciphertext encrypted using Elgamal such that $C = (C_1, C_2) = E(M_i)$ where $i = 1$ or 2. We will now prove that the attack presented in the assignment works. To prove that this attack works, let's examine each case individually.

First we know that $M_1^{\frac{p-1}{2}} \equiv 1 \ (mod\ p)$ and $M_2^{\frac{p-1}{2}} \equiv -1 \ (mod\ p)$ by Euler's Criterion.

Case 1 ($\left(\frac{y}{p}\right) = 1$ and $\left(\frac{C_2}{p}\right) = 1$): In this case, $y^{\frac{p-1}{2}} \equiv 1 \ (mod\ p)$ and $C_2^{\frac{p-1}{2}} \equiv 1 \ (mod\ p)$. Applying substitutions:

$$y^{\frac{p-1}{2}} \equiv g^{\frac{x(p-1)}{2}} \equiv 1 \ (mod\ p) \qquad (1) \text{ - By Euler's criterion}$$

$$\begin{aligned}
C_2^{\frac{p-1}{2}} &\equiv (My^k)^{\frac{p-1}{2}} \\
&\equiv (M(g^x)^k)^{\frac{p-1}{2}} \\
&\equiv (Mg^{xk})^{\frac{p-1}{2}} \\
&\equiv M^{\frac{p-1}{2}} g^{\frac{xk(p-1)}{2}} \\
&\equiv (1) g^{\frac{xk(p-1)}{2}} && \text{- If } M = M_1 \text{ by Euler's criterion} \\
&\equiv (1)(1)^k && \text{- Substitution by (1)} \\
&\equiv 1 \ (mod\ p)
\end{aligned}$$

Thus if Mallory sees case 1, she knows that $M = M_1$.

Case 2 ($\left(\frac{y}{p}\right) = 1$ and $\left(\frac{C_2}{p}\right) = -1$): In this case, $y^{\frac{p-1}{2}} \equiv 1 \ (mod\ p)$ and $C_2^{\frac{p-1}{2}} \equiv -1 \ (mod\ p)$. Applying substitutions:

$$y^{\frac{p-1}{2}} \equiv g^{\frac{x(p-1)}{2}} \equiv 1 \ (mod\ p) \qquad (1) \text{ - By Euler's criterion}$$

$$\begin{aligned}
C_2^{\frac{p-1}{2}} &\equiv (My^k)^{\frac{p-1}{2}} \\
&\equiv (M(g^x)^k)^{\frac{p-1}{2}} \\
&\equiv (Mg^{xk})^{\frac{p-1}{2}} \\
&\equiv M^{\frac{p-1}{2}} g^{\frac{xk(p-1)}{2}} \\
&\equiv (-1) g^{\frac{xk(p-1)}{2}} && \text{- If} M = M_2 \text{ by Euler's criterion} \\
&\equiv (-1)(1)^k && \text{- Substitution by (1)} \\
&\equiv -1 \ (mod\ p)
\end{aligned}$$

Thus if Mallory sees case 2, she knows that $M = M_2$.

Case 3 ($\left(\frac{y}{p}\right) = -1$, $\left(\frac{C_1}{p}\right) = 1$, and$\left(\frac{C_2}{p}\right) = 1$): In this case $y^{\frac{p-1}{2}} \equiv -1 \ (mod\, p)$, $C_1^{\frac{p-1}{2}} \equiv 1 \ (mod\, p)$, $C_2^{\frac{p-1}{2}} \equiv 1 \ (mod\ p)$

$$y^{\frac{p-1}{2}} \equiv g^{\frac{x(p-1)}{2}} \equiv -1 \ (mod\ p) \qquad (1) \text{ - By Euler's criterion}$$

$$C_1^{\frac{p-1}{2}} \equiv g^{\frac{k(p-1)}{2}} \equiv 1 \ (mod\ p) \qquad (2) \text{ - By Euler's criterion}$$

$$\begin{aligned}
C_2^{\frac{p-1}{2}} &\equiv (My^k)^{\frac{p-1}{2}} \\
&\equiv (M(g^x)^k)^{\frac{p-1}{2}} \\
&\equiv (Mg^{xk})^{\frac{p-1}{2}} \\
&\equiv M^{\frac{p-1}{2}} g^{\frac{xk(p-1)}{2}} \\
&\equiv (1)g^{\frac{xk(p-1)}{2}} && \text{- If} M = M_1 \text{ by Euler's criterion} \\
&\equiv (1)(1)^k && \text{- Substitution by (2)} \\
&\equiv 1 \ (mod\ p)
\end{aligned}$$

Case 4 ($\left(\frac{y}{p}\right) = -1$, $\left(\frac{C_1}{p}\right) = 1$, and $\left(\frac{C_2}{p}\right) = -1$): In this case $y^{\frac{p-1}{2}} \equiv -1 \ (mod\ p)$, $C_1^{\frac{p-1}{2}} \equiv 1 \ (mod\ p)$, $C_2^{\frac{p-1}{2}} \equiv -1 \ (mod\ p)$

$$y^{\frac{p-1}{2}} \equiv g^{\frac{x(p-1)}{2}} \equiv -1 \ (mod\ p) \qquad (1) \text{ - By Euler's criterion}$$

$$C_1^{\frac{p-1}{2}} \equiv g^{\frac{k(p-1)}{2}} \equiv 1 \ (mod\ p) \qquad (2) \text{ - By Euler's criterion}$$

$$\begin{aligned}
C_2^{\frac{p-1}{2}} &\equiv (My^k)^{\frac{p-1}{2}} \\
&\equiv (M(g^x)^k)^{\frac{p-1}{2}} \\
&\equiv (Mg^{xk})^{\frac{p-1}{2}} \\
&\equiv M^{\frac{p-1}{2}} g^{\frac{xk(p-1)}{2}} \\
&\equiv (-1)g^{\frac{xk(p-1)}{2}} && \text{- If} M = M_2 \text{ by Euler's criterion} \\
&\equiv (-1)(1)^k && \text{- Substitution by (2)} \\
&\equiv -1 \ (mod\ p)
\end{aligned}$$

Case 5 ($\left(\frac{y}{p}\right) = -1$, $\left(\frac{C_1}{p}\right) = -1$, and $\left(\frac{C_2}{p}\right) = 1$): In this case $y^{\frac{p-1}{2}} \equiv -1 \ (mod\ p)$, $C_1^{\frac{p-1}{2}} \equiv -1 \ (mod\ p)$, $C_2^{\frac{p-1}{2}} \equiv 1 \ (mod\ p)$

$$y^{\frac{p-1}{2}} \equiv g^{\frac{x(p-1)}{2}} \equiv -1 \ (mod\ p) \qquad (1) \text{ - By Euler's criterion}$$

$$C_1^{\frac{p-1}{2}} \equiv g^{\frac{k(p-1)}{2}} \equiv -1 \ (mod\ p) \qquad (2) \text{ - By Euler's criterion}$$

$$\begin{aligned}
C_2^{\frac{p-1}{2}} &\equiv (My^k)^{\frac{p-1}{2}} \\
&\equiv (M(g^x)^k)^{\frac{p-1}{2}} \\
&\equiv (Mg^{xk})^{\frac{p-1}{2}} \\
&\equiv M^{\frac{p-1}{2}} g^{\frac{xk(p-1)}{2}} \\
&\equiv (-1)g^{\frac{xk(p-1)}{2}} && \text{- If} M = M_2 \text{ by Euler's criterion} \\
&\equiv (-1)(-1)^k && \text{- Substitution by (1)} \\
&\equiv 1 \ (mod\ p)
\end{aligned}$$

This holds because $k \in \mathbb{Z}_{p-1}$, which implies k is odd, and $(-1)^k \neq 1$.

Case 6 ($\left(\frac{y}{p}\right) = -1$, $\left(\frac{C_1}{p}\right) = -1$, and $\left(\frac{C_2}{p}\right) = -1$): In this case $y^{\frac{p-1}{2}} \equiv -1 \ (mod\ p)$, $C_1^{\frac{p-1}{2}} \equiv -1 \ (mod\ p)$, $C_2^{\frac{p-1}{2}} \equiv -1 \ (mod\ p)$

$$y^{\frac{p-1}{2}} \equiv g^{\frac{x(p-1)}{2}} \equiv -1 \ (mod\ p) \qquad (1) \text{ - By Euler's criterion}$$

$$C_1^{\frac{p-1}{2}} \equiv g^{\frac{k(p-1)}{2}} \equiv -1 \ (mod\ p) \qquad (2) \text{ - By Euler's criterion}$$

$$\begin{aligned}
C_2^{\frac{p-1}{2}} &\equiv (My^k)^{\frac{p-1}{2}} \\
&\equiv (M(g^x)^k)^{\frac{p-1}{2}} \\
&\equiv (Mg^{xk})^{\frac{p-1}{2}} \\
&\equiv M^{\frac{p-1}{2}} g^{\frac{xk(p-1)}{2}} \\
&\equiv (1)g^{\frac{xk(p-1)}{2}} && \text{- If} M = M_1 \text{ by Euler's criterion} \\
&\equiv (1)(-1)^k && \text{- Substitution by (1)} \\
&\equiv -1 \ (mod\ p)
\end{aligned}$$

This holds because $k \in \mathbb{Z}_{p-1}$, which implies k is odd, and $(-1)^k \neq 1$.

We can see in each of these cases that the legendre symbol calculations depend upon which plaintext has been encrypted. Thus Mallory can assert with certainty which plaintext corresponds to which encryption and the Elgamal encryption scheme is not semantically secure. QED

Problem 5 — An IND-CPA, but not IND-CCA secure version of RSA, 12 marks

Mallory can perform a multiplicative attack on this version of RSA.

Proof. She generates an $X \in \mathbb{Z}_n^*$ where $X^e \neq 1 \ (mod\ n)$. Both n and e are public information so it's easy to perform this step. Then for her chosen ciphertext she submits $C' \equiv CX^e \ (mod\ n)$, where C is the ciphertext given back to her by the oracle. She then proceeds as follows:

$$M' \equiv (C')^d \equiv C^d (X^e)^d \equiv MX \ (mod\ n)$$

In this IND-CPA secure version she knows the values s and t of the ciphertext handed back to her because they're simply concatenated and she knows their respective lengths ($|t| = m$). She would proceed as follows:

$$\begin{aligned} C' &= (s||t \oplus M_1 1) = C \oplus (0^{|s|}||M_1) \qquad \text{- } M' = M_1 \oplus M_1 = 0 \\ &= (r^e \ (mod\ n)||H(r) \oplus M_i \oplus M_1) \\ &= (r^e \ (mod\ n)||H(r)) \end{aligned}$$

(At least she assumes $M_i = M_1$, if it's not then $H(r) = H(r) \oplus M_2 \oplus M_1$ which she will see at the end)

$$\begin{aligned} C' &\equiv CX^e \ (mod\ n) \\ C'^d &\equiv (CX^e)^d \ (mod\ n) \\ &\equiv (r^e \ (mod\ n)||H(r))^d \\ &\equiv ((r^e \ (mod\ n)||H(r) \oplus M_i)X^e)^d \\ &\equiv (r^e \ (mod\ n)||H(r) \oplus M_i)^d X^{ed} \\ &\equiv (r^e \ (mod\ n)||H(r) \oplus M_i)^d X \qquad \text{- Because } ed = 1 + k\phi(n) \equiv 1 \ (mod\ n) \\ &\equiv C^d X \\ &\equiv M_i X \ (mod\ n) \qquad \text{- (Once more assuming } M_i = M_1) \end{aligned}$$

In order for this to work Mallory would need to calculate X. (As $\oplus$ provides binary addition, not multiplication). Therefore she could use the EE algorithm to find the inverse of C such that $CC^{-1} \equiv 1 \ (mod\ n)$ so that $C' \equiv CX^e \ (mod\ n)$ becomes $C'C^{-1} \equiv CC^{-1}X^e \equiv X^e \ (mod\ n)$. At this point Mallory would need to use X^e to find X. This is done by simply calculating $X = (X^e)^{\frac{1}{e}}$. Through the EE algorithm we can find X^{-1} that satisfies $XX^{-1} \equiv 1 \ (mod\ n)$, then she can compute $M_i \equiv M_i X X^{-1} \ (mod\ n)$. If at this point $M_i = M_1$ then she knows that $M_i = M_1$ and that $H(r) = H(r) \oplus M_i \oplus M_1$. Otherwise, $M_i = M_2$, and she will get some form of garbage because $H(r) \oplus M_2 \oplus M_1$ was used instead, so she concludes $C = E_k(M_2)$. $C' \neq C$ because $(s||t \oplus M_1) \neq (s||t)$. Finding modular inverses can be done in polynomial time, therefore this version of RSA is provably not IND-CCA secure. QED

Problem 6 — An attack on RSA with small decryption exponent, 25 marks

(a)

(b)

(c)

(d)

(e)

(f)

Problem 7 — Universal forgery attack on the El Gamal signature scheme, 12 marks)

(a)

(b)

(c)

Problem 9 — Columnar transposition cryptanalysis, 10 marks