Problem 1

a. We construct an adversary as follows: we can efficiently find two strings \(m_0\) and \(m_1\) where \(m_0\) is compressible with the compression function and \(m_1\) is not. Let \(m_0'\) and \(m_1'\) be the compressed strings of \(m_0\) and \(m_1\) respectively, which we know have different lengths. Send these strings to the challenger and receive \(c\). Output 0 if \(|c| = |m_0'| + 128\) and 1 otherwise. We can clearly see that the adversary is efficient and achieves advantage \(|\Pr[EXP(0) = 1] - \Pr[EXP(1) = 1]| = |0 - 1| = 1\). Thus, we have an efficient adversary against this compress-then-encrypt system, and so it isn’t semantically secure.

b. We assume that a message is encrypted by a semantically secure cipher before compression. Because the encryption is semantically secure, we know that the ciphertext is indistinguishable from a random string of the same length. Thus, because the ciphertext is effectively random, compression will have a much reduced effectiveness. Compression algorithms take advantage of redundancy or patterns in strings, which we can assume the encryption will very likely eliminate.

Problem 2

a. Let the nodes \(v_1, ..., v_{\log_2 n}\) be the nodes along the path from the root to the exposed leaf node \(r\). Because the tree is binary, every node \(v_i\) for \(2 \leq i \leq \log_2 n\) has exactly one sibling. Let \(s_2, s_3, ..., s_{\log_2 n}\) be the siblings of the nodes \(v_1, ..., v_{\log_2 n}\). Furthermore, let \(S = \{K_2, ..., K_{\log_2 n}\}\) be the AES keys associated with the nodes \(s_2, ..., s_{\log_2 n}\). The new header will contain the encryption of \(K\) under all keys in \(S\). DVD player \(r\) cannot decrypt the movie because it contains none of the keys in \(S\). However, every other player \(p \neq r\) can decrypt the movie. To prove this, consider the path from the root of the binary tree to leaf node \(p\). Let \(u\) be the highest node along this path that does not appear on the path from the root to \(r\). Then, the key \(K_u\) associated with node \(u\) is in the set \(S\), because the sibling of \(u\) is on the path to \(r\). Furthermore, player \(p\) contains the key \(K_u\), which it can use to decrypt \(K\) and thus decrypt the movie. Thus, player \(r\) cannot decrypt the movie, but all other players can.

b. For \(i = 1, ..., k\) let \(U_i\) be the set of consecutive leaves in the range \([r_i + 1, r_{i+1} - 1]\) where \(r_0 = 0\) and \(r_{k+1} = n + 1\). For any internal node \(v\) in the tree, let \(S_v\) be the set of leaves in
the subtree rooted at v. U_i is covered by internal nodes v_1, ..., v_b is U_i = \bigcup_{j=1}^{b} S_{v_j}. We show below that each set U_i can be exactly covered by a set of at most 2\log_2 n internal nodes. This means that to cover exactly all sets U_0, ..., U_r, we need at most c = 2(r + 1)\log_2 n internal nodes. Let v_1, ..., v_c be the internal nodes needed to cover all of U_0, ..., U_r. If we encrypt K using the keys K_v associated with each of these nodes, then all players other than those in R can recover K while the players in R cannot. This shows that using a header containing at most 2(r + 1)\log_2 n ciphertexts, we can revoke all players in R without affecting any of the other players.

It remains to show that given a set of consecutive leaves U in some range [a, b] it is possible to exactly cover U using at most 2\log_2 n internal nodes. Let u_1 be the highest node in the tree such that the subtree rooted at u_1 has leaf a as its left most leaf, and all the leaves in the subtree are contained in [a, b]. Let v_1 be the highest node in the subtree rooted at v_1 that has leaf b as its right most leaf. Now, for i = 2, ..., \log_2 n define u_i to be the right sibling of the parent of u_{i-1} (the right sibling of a node w is the node at the same height as w which is immediately on the right of w). Similarly, define v_i to be the left sibling of the parent of v_{i-1}. Let j be the smallest value so that u_j = v_j or that u_j, v_j are adjacent siblings in the tree. Then it is easy to see that u_1, ..., u_j, v_j, ..., v_1 is an exact cover of [a, b]. This covering set contains at most 2\log_2 n nodes as required.

**Problem 3**

a. We have Pr[W_0] = 1 = Pr[W_1]. Thus, Adv(A_1) = |1 - 1| = 0.
   - We have Pr[W_0] = 0.5 = Pr[W_1]. Thus, Adv(A_2) = |0.5 - 0.5| = 0.
   - We have Pr[W_0] = 0.5 and Pr[W_1] = 0. Thus, Adv(A_2) = |0.5 - 0| = 0.5.
   - We have Pr[W_0] = 0.5 and Pr[W_1] = 1. Thus, Adv(A_2) = |0.5 - 1| = 0.5.
   - We have Pr[W_0] = 0.5 + 0.5 \cdot 0.5 = 0.75 and Pr[W_1] = 0.5. Thus, Adv(A_2) = |0.75 - 0.5| = 0.25.

b. Let X be the value that the adversary receives from the challenger. This is the only information the adversary receives, so his action can be modeled as a probabilistic response conditioned on X. That is, P_H = Pr[\text{output 1}|X = HEADS] and P_T = Pr[\text{output 1}|X = TAILS]. We can now break the advantage into cases:

\[
Adv = |Pr[W_0] - Pr[W_1]|
= |(0.5P_H + 0.5P_T) - (0 \cdot P_H + 1 \cdot P_T)|
= |0.5(P_H + P_T) - P_T|
= 0.5|P_H - P_T|
\]

The maximum value of P_H - P_T is 1, and so the advantage is at most 0.5.
Problem 4

a. We prove the system is semantically secure by contrapositive. That is, we will show that if system $E_1$ is insecure, then $E$ is also insecure as well, which is impossible. Thus, we assume that system $E_1$ is insecure, meaning there exists an adversary $B$ on $E_1$ that achieves non-negligible advantage. We construct an adversary $A$ that achieves the same advantage on system $E$. $B$ sends messages $m_0$ and $m_1$ to $A$, who forwards that to its challenger for $E$. $A$ receives $c$ and prepends a 0 to it to obtain $c'$, then sends $c'$ to $B$. $A$ then outputs what $B$ outputs. $B$ effectively sees $A$ as its challenger in this case and can distinguish which of $m_0$ and $m_1$ was encrypted, as $A$ has transformed $c$ into the string that $B$ would have seen from its challenger. Thus, because $A$ outputs the same bit as $B$, they achieve the same advantage, implying that $A$ is insecure and completing our contrapositive proof.

b. We present an attack on the system to show that it is not semantically secure. Consider an adversary $A$ that submits $m_0$ of odd parity and $m_1$ of even parity to the challenger, receiving back $c$. $A$ then outputs 0 if the last bit is 1 and 1 otherwise. $A$ is clearly efficient because it calculates parities in the beginning with $O(L)$ time and does a simple check of the last bit of $c$ in $O(1)$ time. $A$ will also only ever output 0 if $m_0$ was encrypted and 1 if $m_1$ was encrypted, and so it achieves an advantage of $Adv = |0 - 1| = 1$. We have constructed an adversary with non-negligible advantage, and so $E_2$ is not semantically secure.

c. We prove the system is semantically secure by contrapositive. That is, we will show that if system $E_3$ is insecure, then $E$ is also insecure as well, which is impossible. Thus, we assume that system $E_3$ is insecure, meaning there exists an adversary $B$ on $E_3$ that achieves non-negligible advantage. We construct an adversary $A$ that achieves the same advantage on system $E$. $B$ sends messages $m_0$ and $m_1$ to $A$, who forwards that to its challenger for $E$. $A$ receives $c$ and reverses it to obtain $c'$, then sends $c'$ to $B$. $A$ then outputs what $B$ outputs. $B$ effectively sees $A$ as its challenger in this case and can distinguish which of $m_0$ and $m_1$ was encrypted, as $A$ has transformed $c$ into the string that $B$ would have seen from its challenger. Thus, because $A$ outputs the same bit as $B$, they achieve the same advantage, implying that $A$ is insecure and completing our contrapositive proof.

d. We prove the system is semantically secure by contrapositive. That is, we will show that if system $E_4$ is insecure, then $E$ is also insecure as well, which is impossible. Thus, we assume that system $E_4$ is insecure, meaning there exists an adversary $B$ on $E_4$ that achieves non-negligible advantage. We construct an adversary $A$ that achieves the same advantage on system $E$. $B$ sends messages $m_0$ and $m_1$ to $A$, who reverses them to obtain $m'_0$ and $m'_1$, then forwards them to its challenger for $E$. $A$ receives $c$ and forwards it to $B$. $A$ then outputs what $B$ outputs. $B$ effectively sees $A$ as its challenger in this case and can distinguish which of $m_0$ and $m_1$ was encrypted, and $A$ is playing the same game with reversed inputs. Thus, because $A$ outputs the same bit as $B$, they achieve the same advantage, implying that $A$ is insecure and completing our contrapositive proof.
Problem 5

a. We construct an adversary that can break the PRF with advantage $1 - \frac{1}{2^n}$.

- Query $x = y = 0^n$ and receive $z = G(k, (x, y))$, where $G$ is either $F_1$ or a random function.
- Output $b=1$ if $z = 0^n$ and 0 otherwise.

$F_1(k, (0^n, 0^n)) = F(k, 0^n) \oplus F(k, 0^n) = 0^n$. The advantage is thus:

$$|Pr[EXP(0) = 1] - Pr[EXP(1) = 1]| = \left| \frac{1}{2^n} - 1 \right| = 1 - \frac{1}{2^n}$$

which is clearly non-negligible.

b. We prove by contradiction, assuming that we have an efficient adversary $A_2$ that can break $F_2$ with non-negligible advantage. We construct an adversary $B$ that breaks $F$ with the same advantage:

- Start running adversary $A_2$.
- Whenever $A_2$ issues a query for $x$, forward $x$ to the challenger. The challenger replies with a value $y$. Send $y \oplus x$ to $A_2$.
- Output what $A_2$ outputs.

We claim that $B$ perfectly simulates what $A_2$ expects when interacting with a PRF challenger for $F_2$.

1. If $b = 0$ (PRF), on a query $x$, $A_2$ receives from $B$ the value $F(k, x) \oplus x = F_2(k, x)$.
2. If $b = 1$ (random function), on a query $x$, $A_2$ receives from $B$ a truly random value $y \oplus x$. This is because for any truly random $y$, $y \oplus x$ is truly random for any $x$.

In both cases, $A_2$'s view of the PRF game is consistent with what it would expect had it been interacting with the real challenger for $F_2$. We conclude then that $B$'s distinguishing advantage is equal to that of $A_2$, which is non-negligible. However, this is a contradiction, so we conclude that $A_2$ cannot exist and thus, $F_2$ is secure.

Problem 6

a. We will construct an adversary that wins the CPA security game against $E$ using $2^{\ell/2}$ queries. Choose $2^{\ell/2} + 1$ distinct messages $m_0, m_1 \ldots m_{\ell/2}$. Now, make $2^{\ell/2}$ queries of the form $(m_0, m_1), (m_0, m_2) \ldots (m_0, m_{2^{\ell/2}})$. If the any of the responses have the same value, output 1. Otherwise, output 0.

We now show that this adversary obtains advantage $\approx 1/2$. If $b = 0$, we have been getting encryptions of the same message $m_0$ from the challenger. Recall that there since
we have an $\ell$-bit nonce in $E$, there are only $\ell$ possible encryptions of any one message. Therefore, by the birthday paradox, the responses will contain at least one collision with a probability of about $1 - \frac{1}{\sqrt{\ell}} = 0.393 \ldots$. So, $\Pr[\text{EXP}(0) = 1] \approx 1/2$.

If $b = 1$, we have been getting encryptions of $m_1 \ldots m_{2^{\ell/2}}$ from the challenger; since the messages were distinct, it must be true that the encryptions are distinct (otherwise, decryption would fail). So, if $b = 1$, the probability we output 1 is 0. So, $\Pr[\text{EXP}(1) = 1] = 0$. Then, our adversary has a non-negligible advantage of $\approx 1/2$.

b. In part A, we showed that an adversary can break the CPA security game against an encryption scheme with an $\ell$-bit nonce with $2^{\ell/2}$ queries. Recall that in CBC mode, we have an ‘initialization vector’ that serves as our nonce and is the same size as the block size. Since the nonce must be included in the ciphertext, the total ciphertext size is $m + \ell$ for an input message of length $m$. Then, CBC mode is a scheme that is susceptible to the attack outlined in part (a), and an adversary can get advantage $\approx 1/2$ in the CPA security game using only $2^{\ell/2}$ queries.

Problem 7

a. Consider the one time pad. This cipher, defined $E(k, m) = k \oplus m$, is semantically secure, as shown in class. However $E_2(k, m) = k \oplus k \oplus m = m$ is clearly not semantically secure. A stream cipher is another correct answer.

b. Assume that $A$ is an attacker on $E_2$ with non-negligible advantage. We can construct an adversary $B$ that can gain the same advantage on $E$ as follows.

Have $A$ start their attack on $E_2$. Since we’re playing the CPA security game, $A$ will make multiples queries consisting of pairs of messages $(m_{0,i}, m_{1,i})$ to $B$, where $i$ indicates which query each message is part of. For each query, have $B$ pass the pair unmodified to the challenger and receive $E(k, m_{b,i})$, where $b$ is either 0 or 1. Since $E_2$ involves two encryptions, have $B$ send $(E(k, m_{b,i}), E(k, m_{b,i}))$ back to the challenger and receive $E(k, E(k, m_{b,i})) = E_2(k, m_{b,i})$. Send this value $E_2(k, m_{b,i})$ to $A$. With this scheme, $B$ emulates a challenger for $E_2$ with respect to $A$. After some number of queries, $A$ will output their guess for the value of $b$. Observe the value that $A$ outputs and have $B$ output that value. This makes the advantage of $B$ on $E$ equal to the advantage on $A$ on $E_2$, which we assume is non-negligible. This contradicts the fact that $E$ is CPA secure, so our assumption that $A$ exists was incorrect. Thus $E_2$ is CPA secure, as required.