Problem 1. Let $m_0$ be the message attack at dawn and $m_1$ be the message attack at dusk, and let $c_0, c_1$ be the corresponding ciphertexts. Since the message is encrypted using a one-time pad, say, $p$, we have:

$$c_0 = m_0 \oplus p$$

So we can obtain the one-time pad by XORing the ciphertext with the plaintext:

$$p = c_0 \oplus m_0$$

and, from there, it is easy to encrypt the new message:

$$c_1 = m_1 \oplus p = m_1 \oplus (c_0 \oplus m_0) = c_0 \oplus (m_0 \oplus m_1)$$

by XORing the first ciphertext with the XOR of the two plaintexts. Since the two plaintexts are identical until the last three characters, their XOR is zero until the last three characters, which are $\text{awn} \oplus \text{usk} = 61776e \oplus 75736b = 140405$. Then:

$$09e1c5f70a65ac51626bc3d25f28 \oplus 00 \cdots 00140405 = 09e1c5f70a65ac51626bc3c65b2d.$$ 

Problem 2. (a) Alice can first pick a random key $K$ to encrypt the message $M$, and then she encrypts the key $K$ with both $K_{AB}$ and $K_{AC}$. In other words, Alice does the following:

$$E_{K_{AB}}[E_{K_{AC}}(K)] \| E_K(M)$$

header

(b) Similar to Part (a), Alice does the following:

$$E_{K_{AB}}[E_{K_{AC}}(K)] \| E_{K_{AB}}[E_{K_{AD}}(K)] \| E_{K_{AC}}[E_{K_{AD}}(K)] \| E_K(M)$$

header

(c) It is straightforward to generalize the above scheme to the case where any $k$ out of the $n$ recipients can decrypt, but any $k-1$ cannot. The length of the header would be $\binom{n}{k} = n!/k!(n-k)!$. Note that we have the inequalities:

$$\left(\frac{n}{k}\right)^{k} \leq \left(\frac{n}{k}\right) \leq \left(\frac{ne}{k}\right)^{k}$$

This shows that the proposed solution scales poorly.

Problem 3. (a) Let nodes $v_1, \ldots, v_{\log_2 n}$ be the nodes along the path from the root of the tree to leaf node number $r$. Since the tree is binary every node $v_i$ for $i = 2, \ldots, \log_2 n$ has exactly one sibling. Let
$u_2, u_3, \ldots, u_{\log_2 n}$ be the siblings of nodes $v_2, \ldots, v_{\log_2 n}$. Let $K_2, \ldots, K_{\log_2 n}$ be the AES keys associated with the nodes $u_2, \ldots, u_{\log_2 n}$. The header will contain the encryption of the content-key $K$ under all keys $\mathcal{K} = \{K_2, \ldots, K_{\log_2 n}\}$. Clearly player number $r$ cannot decrypt the movie since it does not have any of the keys in $\mathcal{K}$. However, any other player $t \neq r$ can decrypt the movie. To see this, consider the path from the root of the binary tree to leaf node $t$. Let $u$ be the highest node along this path that does not appear on the path from the root to leaf node $r$. Then the key $K_u$ associated with node $u$ is in the set $\mathcal{K}$ (since the sibling of $u$ is a on the path to $r$). Furthermore, player $t$ contains the key $K_u$ and it can therefore obtain the content-key and then the movie. Hence, player $r$ cannot play the movie, but all other players can.

(b) For $i = 1, \ldots, k$ let $U_i$ be the set of consecutive leaves in the range $[r_i + 1, r_{i+1} - 1]$ (we are assuming $r_0 = 0$ and $r_k + 1 = n + 1$). For an internal node $v$ in the binary tree let $\mathcal{S}_v$ be the set of leaves in the subtree rooted at $v$. We say that a set of leaves $U_i$ is exactly covered by internal nodes $v_1, \ldots, v_\ell$ if $U_i = \mathcal{S}_{v_1} \cup \cdots \cup \mathcal{S}_{v_\ell}$. We show below that each set $U_i$ can be exactly covered by a set of at most $2 \cdot \log_2 n$ internal nodes. This means that to exactly cover all sets $U_0, \ldots, U_r$ we need at most $c = 2(r + 1)\log_2 n$ internal nodes. Let $v_1, \ldots, v_\ell$ be the internal nodes needed to cover all of $U_0, \ldots, U_r$. If we encrypt the content-key using the keys $K_v$ associated with each of these nodes then all players other than those in $R$ can recover the content-key while the players in $R$ cannot. This shows that using 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 $2 \cdot \log_2 n$ internal nodes. Let $u_1$ be the highest node in the tree so that the subtree rooted at $u_1$ has leaf $a$ as its left most leaf, and all of 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 $i = 2, \ldots, \log_2 n$ 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 to 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 $u_1, u_2, v_1, v_2, \ldots, v_{j+1}$ is an exact cover of $[a, b]$. This covering set contains at most $2 \cdot \log_2 n$ nodes as required.

**Problem 4. (a)**

- We have $\Pr[W_0] = 1 = \Pr[W_1]$. Thus, $\text{Adv}(A_1) = |1 - 1| = 0$.
- We have $\Pr[W_0] = \frac{1}{2} = \Pr[W_1]$. Thus, $\text{Adv}(A_2) = |\frac{1}{2} - \frac{1}{2}| = 0$.
- We have $\Pr[W_0] = \frac{1}{2}$ and $\Pr[W_1] = 0$ so $\text{Adv}(A_3) = |\frac{1}{2} - 0| = \frac{1}{2}$.
- We have $\Pr[W_0] = \frac{1}{2}$ and $\Pr[W_1] = 1$ so $\text{Adv}(A_4) = |\frac{1}{2} - 1| = \frac{1}{2}$.
- We have $\Pr[W_0] = \frac{1}{2} + \frac{1}{2} = \frac{3}{4}$ and $\Pr[W_1] = \frac{1}{2}$, so $\text{Adv}(A_5) = |\frac{3}{4} - \frac{1}{2}| = \frac{1}{4}$.

Observe that any adversary that ignores the information from the coin toss has advantage 0.

(b) Let $X$ be the value that the adversary receives from the challenger. This is the only information that the adversary receives, so his/her actions can only be be a (probabilistic) response conditioned on $X$. That is, the adversary’s strategy is determined by selecting the following independent values:

- $P_H = \Pr[\text{output } 1 \mid X = \text{HEADS}]$
- $P_T = \Pr[\text{output } 1 \mid X = \text{TAILS}]$
We can now break the advantage into cases:

\[
\text{Adv} = \abs{Pr[W_0] - Pr[W_1]} \\
= \abs{\left(\frac{1}{2}P_H + \frac{1}{2}P_T\right) - (0 \cdot P_H + 1 \cdot P_T)} \\
= \abs{\frac{1}{2}(P_H + P_T) - P_T} \\
= \frac{1}{2}|P_H - P_T|
\]

Since \(P_H, P_T \in [0,1]\) are probabilities, the maximum difference between them is 1, and the advantage is at most 1/2. Note also that the adversaries \(A_3\) and \(A_4\) from part (a) achieve this, so in fact 1/2 is in fact the maximum possible advantage.

**Problem 5.**

XOR-ing a message with an even number of 1-bits does not change the parity of the message. Therefore, the adversary can select for \(m_0\) a message of even parity (e.g. \(0^n\)) and for \(m_1\) a message of odd parity (e.g. \(0^{n-1}1\)). If the challenger outputs a ciphertext of even parity, the adversary returns 1. Otherwise, the adversary returns 0.

The adversary is obviously efficient because computing parity is \(O(L)\).

The adversary can perfectly distinguish between the two experiments and thus has an advantage of 1:

\[
\text{Adv} = \abs{Pr[W_0] - Pr[W_1]} \\
= \abs{0 - 1} \\
= 1
\]

**Problem 6.** (a) Let \(n\) be the block size for the encryption scheme, and let \(F\) be the block cipher used for CBC-encryption. For instance, \(n = 128\) for AES-128. Construct the following CPA-adversary:

1. Send a single-block query \((0^n, 0^n)\) to the challenger. The challenger replies with a ciphertext \(c_0\).
2. Send a single-block query \((c_0, c_0)\) to the challenger. The challenger replies with a ciphertext \(c_1\).
3. Send a single-block query \((c_1, c')\) where \(c' \neq c_1\) is an arbitrary element of the message space. The challenger replies with a ciphertext \(c_2\).
4. Output 1 if \(c_2 = c_1\).

Clearly, this is an efficient adversary. We now show that it achieves advantage 1 in the CPA security game. Consider the second CPA-query \((c_0, c_0)\). By construction, the challenger will use \(c_0\) as the IV used to encrypt the query message \(c_0\), and so, we have that \(c_1 = F(k, c_0 \oplus c_0) = F(k, 0^n)\). Note that \(c_1\) will be used as the IV in the final CPA-query. Consider the challenger’s response to the third CPA query:
• In World 0, the challenger will encrypt $c_1$ using the IV $c_1$ which yields $c_2 = F(k, c_1 \oplus c_1) = F(k, 0^n) = c_1$. In this case, the adversary outputs 1 with probability 1.

• In World 1, the challenger will encrypt $c' \neq c_1$ using the IV $c_1$ which yields $F(k, c_1 \oplus c')$. Since $c_1 \neq c'$, $c_1 \oplus c' \neq 0^n$, and since $F$ is a PRP, $c_2 \neq c_1$. Thus, the adversary outputs 1 with probability 0.

We conclude that the adversary has distinguishing advantage exactly 1.

(b) Apply the PRP to the last block of the previous ciphertext using a new key before using it as the IV. In order to predict the IV, the adversary would have to guess the output of a PRP. Security of the PRP implies that the adversary is able to do so with negligible probability.

(c) Consider the following CPA-adversary:

1. Send a single-block query $(0^n, 0^n)$ to the challenger. The challenger replies with $(IV, c_0)$.
2. Send a single-block query $(0^n, 1^n)$ to the challenger. The challenger replies with $(IV', c_1)$.
3. The adversary outputs 1 if $IV = IV'$ and $c_0 = c_1$, and 0 otherwise.

Again, the adversary is clearly efficient. We argue that the advantage of this adversary is $\frac{1}{2^n}$. The IV is chosen independently and uniformly at random in both worlds so the probability that $IV = IV'$ is $\frac{1}{2^n}$. In World 0, if $IV = IV'$, then $c_0 = c_1$, so the adversary outputs 1 with probability $\frac{1}{2^n}$. In World 1, if $IV = IV'$, then $c_0 \neq c_1$, so the adversary outputs 1 with probability 0. We conclude that the adversary’s advantage is exactly $\frac{1}{2^n}$.

Problem 7. (a) A 1-query adversary can break the PRF with advantage $1 - 2^{-n}$ using the following strategy in the PRF security game:

• 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:

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

which is clearly non-negligible for any $n > 0$.

(b) Suppose we have an efficient adversary $A_2$ that can break $F_2$ with non-negligible advantage. We can construct an adversary $B$ that breaks $F$ as follows:

1. Start running adversary $A_2$.
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 $A_2$.
3. Output the same guess $b$ as $A_2$ outputs.

We claim that $B$ perfectly simulates the view $A_2$ expects when interacting with a PRF challenger for $F_2$:

• In World 0 (PRF world), on a query $x$, $A_2$ receives from $B$ the value $F(k, x) \oplus x = F_2(k, x)$. 


• In World 1 (random), 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. But this contradicts the assumption that \( F \) is a secure PRF, so we conclude that \( A_2 \) cannot exist, or equivalently, that \( F_2 \) is secure.