1. Let $F$ be a secure PRP with block length $\lambda$. Consider the encryption algorithm below:

$$
\text{Enc}(k, m) : \\
\begin{align*}
  r &\leftarrow \{0, 1\}^\lambda \\
  x &:= F(k, r \oplus m) \\
  \text{return } (r, x)
\end{align*}
$$

Write the decryption algorithm. Then show that the scheme does not have CCA security. (Describe a distinguisher and compute its advantage.)

2. Let $E$ be a CPA-secure encryption scheme and $M$ be a secure MAC. Show that the following encryption scheme (called encrypt & MAC) is not CCA-secure:

$$
\begin{align*}
\text{E&M.KeyGen:} & \quad k_e \leftarrow E.\text{KeyGen} \\
& \quad k_m \leftarrow M.\text{KeyGen} \\
& \quad \text{return } (k_e, k_m)
\end{align*}
$$

$$
\begin{align*}
\text{E&M.Enc((k_e, k_m), m):} & \quad c \leftarrow E.\text{Enc}(k_e, m) \\
& \quad t := M.\text{MAC}(k_m, m) \\
& \quad \text{return } (c, t)
\end{align*}
$$

$$
\begin{align*}
\text{E&M.Dec((k_e, k_m), (c, t)):} & \quad m := E.\text{Dec}(k_e, c) \\
& \quad \text{if } t \neq M.\text{MAC}(k_m, m): \\
& \quad \quad \text{return } \text{err} \\
& \quad \quad \text{return } m
\end{align*}
$$

Describe a distinguisher and compute its advantage.

3. When a user creates a new account at a website, they receive a browser cookie containing $(d, \text{MAC}(k, H(d)))$, where: $d$ is a string of the form "username|timestamp", $H$ is a hash function, and $k$ is the website’s secret key. The timestamp reflects the time that the cookie/account was created, encoded in format yyyymmdd; usernames must consist of entirely lowercase characters a-z. When a user connects to the website, the website checks that the MAC is valid and that the timestamp is in the past. A new user can request an account for any (available) username.

In general $\text{MAC}(k, H(d))$ is a secure MAC of $d$, but suppose the website inexplicably uses $H(d) =$ first 5 bytes (40 bits) of MD5$(d)$. I already have an account on the server with username mikero, and I won’t let you see my authentication cookie.

Find and show me a username (other than mikero) that you can register for on this homework’s due date (Feb 26) which will allow you to authenticate as mikero. Describe the MAC forgery you have in mind and show how you found it, and why it takes the amount of time that it does.

Note: I chose the parameters for this problem so that you will not be able to solve this problem if you take the wrong approach. You can easily do $2^{20}$ work before the due date but...
probably not $2^{40}$. My solution typically takes between 5 and 10 seconds to find a forgery. If yours hasn’t stopped after a few minutes, then you are probably doing it wrong.

*Tip:* You should be able to find an MD5 library implementation in your language of choice. In lieu of that, you can use the Linux command line as follows:

```
$ echo -n "mikero|20180219" | md5sum
014faa2a897a42e9af9d0b8f25087e8f -
$ echo -n "mikero|20180219" | md5sum | cut -c1-10
014faa2a89
```

echo -n sends its argument to md5sum without adding a trailing newline character. cut -c1-10 returns the first 10 characters of md5sum’s output. Since md5sum returns the hex-encoded hash, the first 10 hex characters correspond to the first 5 bytes.

grad. Show that a scheme has CCA$^*$ security if and only if the following two libraries are interchangeable. That means a security proof in both directions.

<table>
<thead>
<tr>
<th>$L_{\Sigma}^{\text{left}}$</th>
<th>$L_{\Sigma}^{\text{right}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k \leftarrow \Sigma.\text{KeyGen}$</td>
<td>$k \leftarrow \Sigma.\text{KeyGen}$</td>
</tr>
<tr>
<td>$D := \text{empty assoc. array}$</td>
<td>$D := \text{empty assoc. array}$</td>
</tr>
<tr>
<td>$\text{CHALLENGE}(m \in \Sigma.M)$:</td>
<td>$\text{CHALLENGE}(m \in \Sigma.M)$:</td>
</tr>
<tr>
<td>\hspace{1em} return $\Sigma.\text{Enc}(k, m)$</td>
<td>\hspace{1em} $c \leftarrow \Sigma.C(</td>
</tr>
<tr>
<td>\hspace{1em} $\text{DEC}(c \in \Sigma.C)$:</td>
<td>\hspace{1em} $D[c] := m$</td>
</tr>
<tr>
<td>\hspace{2em} return $\Sigma.\text{Dec}(k, c)$</td>
<td>\hspace{2em} return $c$</td>
</tr>
<tr>
<td>\hspace{2em} if $D[c]$ exists: return $D[c]$</td>
<td>\hspace{2em} else: return $\Sigma.\text{Dec}(k, c)$</td>
</tr>
</tbody>
</table>

*Note:* In $L_{\Sigma}^{\text{left}}$, the adversary can obtain the decryption of any ciphertext via $\text{DEC}$. In $L_{\Sigma}^{\text{right}}$, the $\text{DEC}$ subroutine is “patched” (via $D$) to give reasonable answers to ciphertexts generated in $\text{CHALLENGE}$. 