Activity 1

In this activity, we will try to understand security game notation and some intuition of what is going on. Let’s go back to the safe example: There is a safe with a three-number combination lock. There are many possible notions of security, but one possible formal security game is the following:

Let \( D = \{1, 2, 3, \ldots, 60\} \) be the set of integers between 1 and 60. Let \( A \) be an adversary.

**Game SAFE\(_D\)**

```plaintext
procedure Initialize
a \leftarrow D; b \leftarrow D; c \leftarrow D

procedure TryCombo(x, y, z)
If \((x, y, z) \notin D \times D \times D\) then return \(\bot\)
If \((x = a) \text{ and } (y = b) \text{ and } (z = c)\) then return true
Else return false

procedure Finalize(\(\alpha, \beta, \gamma\))
If \((\alpha, \beta, \gamma) \notin D \times D \times D\) then return false
If \((\alpha = a) \text{ and } (\beta = b) \text{ and } (\gamma = c)\) then return true
Else return false
```

**Definition:** \(\text{Adv}^{\text{safe}}_{\text{SAFE}_D} = \Pr[\text{SAFE}_D^A \Rightarrow \text{true}]\).

In the above, we have the game \(\text{SAFE}_D\) and the advantage \(\text{Adv}^{\text{safe}}_{\text{D}}\) of an adversary \(A\), which captures how successful \(A\) is in winning the game.

**Question 1:** With your group, decide which parts of this game are initialization, extra adversary capabilities, and the adversary’s win condition. Come up with an explanation in words of those three components — what is happening in the real world?

**Question 2:** In reality, a combination lock will often open if you are within one of each number. How would you modify game \(\text{SAFE}_D\) to capture this? Should you modify \(\text{TryCombo}, \text{Finalize}\), both, or neither?

**Question 3:** Is this more similar to a target or consistent key-recovery notion? How would you modify the game to capture the other?

Activity 2

In this activity we’ll illustrate the difference between target and consistent key recovery, and the impact of the number of queries on each of these security notions, via the game Wordle. (Note that in Wordle you make a different kind of query than you do in the actual target/consistent key recovery game, so this is for illustration only.)

In Wordle there is a secret five-letter word you are trying to find. You make guesses (queries) and learn which letters in your query also appear in the secret word (indicated here with `BOLD`), and also which letters of your query appear in the secret word in the same position as in your guessed word (indicated here with `BOLD AND UNDERLINE`).

Here are three queries.
**Activity 3**

In this activity you’ll convert a pseudocode description of an algorithm into a diagram and reason about how information flows through it. This is a useful first step for many (if not most) homework and exam problems in this class.

For the left diagram above, XOR is symmetric: if you know any two of \(a, b,\) and \(c\) in the diagram, you can XOR them to get the third.

For the right diagram above, block cipher encryption (the \(E\) box) is not symmetric, but block cipher decryption \(E_{K^{-1}}\) lets you go from \(B\) back to \(A\) (assuming you use the same key \(K\)).

**Question 1:** Let \(E : \mathcal{K} \times \mathcal{K} \rightarrow \mathcal{K}\) be a block cipher with equal key, message, and output spaces. We define a new block cipher \(F\) with keyspace \(\mathcal{K} \times \mathcal{K} \times \mathcal{K}\) as follows:

\[
F((K_1, K_2, K_3), M) = K_3 \oplus E_{K_2}(M \oplus K_1)
\]

Draw \(F\) as a diagram.

**Question 2:** Now suppose you are asked how to decrypt a ciphertext encrypted with \(F\). (Assume you know the key.) In this situation, which values in your diagram are known, which are unknown, which (if any) can you choose, and which values are you trying to find out?

**Question 3:** Given the values you know, what other values in your diagram can you compute? Can you decrypt the ciphertext?