Problem 1: Suppose you want to design a client-side protection tool against cross-site request forgery using an HTTP proxy or a browser plugin. What would be the main issues to solve and what security checks might your tool perform? It is important that your defense not prevent "normal" webpages from rendering correctly in the browser.

Problem 2: Prof. Iris Itoai, a biometrics researcher, has developed a remarkable new biometric authentication system that can scan retinas using an ordinary mobile phone when a user takes a selfie. In this problem, you’ll figure out how to use her scheme to convert retinas into cryptographic keys.

For a given user $j$, her system identifies a sequence $V_j = \{v_{1,j}, v_{2,j}, \ldots, v_{k_j,j}\}$ of veins, where $k_j \geq 256$. When scanning an eye, Iris’s system assigns a blood flow score in the range $[0,7]$ to every vein $v_{i,j}$. A scan in Iris’s system for user $j$ thus takes the form of a sequence $S = \{s_i\}_{i=1}^{k_j}$, i.e., a sequence of blood-flow readings.

When a given user registers, i.e., generates a template, the obtained blood flow scores are random, independent values. That is, a template takes the form of a sequence $T_j = \{t_{i,j}\}_{i=1}^{k_j}$, where the scores $\{t_{i,j}\}$ are distributed independently and uniformly at random in $[0,7]$; we will write $T_j \leftarrow \$\$ to denote the (random) generation of a fresh template for user $j$.

Unfortunately, the Itoai retina-scan system is highly error prone. While it consistently detects all veins and orders them consistently, blood flow scores can
vary from authentication to authentication for a given user. When a user undergoes a retina scan, any blood flow score may be off by 1; the only guarantee is that for every $i$, it is the case that $s_i = (t_{i,j} + e) \mod 8$, for integer $|e| \leq 1$. It is thus possible that all of a user’s scanned blood-flow scores are wrong!

Design a scheme that converts a user’s Itoai-type retina scan into a 128-bit cryptographic key. The scheme involves storing only a “processed template” for a user and using this, along with the user’s retina scan, to rederive her key. The scheme will consist of two algorithms:

- **Register$(T_j)$** is a (possibly randomized algorithm) takes in a template for a given user $j$ and outputs a “processed template” $(\sigma_j, k_j)$; and

- **KeyExtract$(S, (\sigma_j, k_j))$** is a deterministic algorithm that takes as input a retina scan and processed template and outputs either a key $\kappa \in \{0, 1\}^{128}$ or a failure symbol $\bot$.

Your scheme should have a few properties:

1. The 128-bit key assigned to a new user, i.e., $\kappa_j \leftarrow \text{KeyExtract}(T_j, \text{Register}(T_j))$ for $T_j \leftarrow s$, should be distributed almost uniformly at random.

2. Despite the high error rate, given a scan $S$ from a valid user $j$, recovery of her key $\kappa_j$ should be perfect, i.e., it is always the case that $\kappa_j \leftarrow \text{KeyExtract}(S, (\sigma_j, k_j))$.

3. It should be hard for an adversary $A$ that compromises the processed template $(\sigma_j, k_j)$ to guess $\kappa_j$. Specifically, for any $A$, we should have:

   $$\Pr[A(\sigma_j, k_j) = \kappa_j \mid T_j \leftarrow s, \sigma_j \leftarrow \text{Register}(T_j), \kappa_j \leftarrow \text{KeyExtract}(T_j, (\sigma_j, k_j))] \leq \epsilon$$

   for negligible $\epsilon$ (not much larger than $2^{-128}$).

Specify a pair of algorithms (Register, KeyExtract) and briefly explain why your scheme achieves Properties 2. and 3. above. (Formal proof isn’t required.)

Hints: You will want to consult the Dodis et al. ’04 paper assigned as reading. Obviously, an error-correcting code will be required. A simple repetition code will suffice here. If you use a cryptographic hash function in your construction, you may assume the random oracle model or make other reasonable, heuristic assumptions about its behavior.
Problem 3: In their “How to Shop for Free Online” paper, Wang et al. describe vulnerabilities in several cashier-as-a-service protocols. For each of these protocols, explain how its design violates one or more of the principles listed by Abadi and Needham in their paper on “Prudent Engineering Practice for Cryptographic Protocols.”

Problem 4: Suppose we wish to design countermeasures for the attacks discussed in the Boneh-Brumley and Yarom-Faulkner papers.

1. Does using blinding of exponentiation with a fixed random multiplier (instead of randomized for each exponentiation) suffice to prevent the Brumley-Boneh timing attack? What about the Flush+Reload attack? Explain why or why not in each case.

2. Explain the simplest configuration-based mechanism (meaning, one need not change software or hardware) by which one can prevent the Flush+Reload attack. Explain the Prime+Probe protocol for inferring cache usage. Does the countermeasure for Flush+Reload prevent a Prime+Probe attack?

3. These attacks focus on victims running cryptographic processes. Provide several examples of non-cryptographic targets for which remote timing and/or cache-based side-channels would enable effective attacks.