Suppose that Bob discovers a polynomial-time algorithm that, given $E_A(M)$ for any (possibly random) M, has a 1% probability of returning "Success: M" and a 99% probability of returning "Sorry, I failed to break it for this input". (The same input produces the same output, i.e., if the algorithm fails to obtain M for a particular $E_A(M)$ then it will fail again if given that same $E_A(M)$ as input.) Here $E_A(M)$ denotes encryption of M with Alice's public key in a cryptosystem that has the multiplicative property, i.e., $E_A(M * R) = E_A(M) * E_A(R)$.

Explain how Bob can use this algorithm to construct another algorithm that runs in polynomial time with extremely high probability (close to 1) and, when it terminates, it *always* obtains M from $E_A(M)$.

Alice has a piece of information that is valuable for the next 3 hours. She offers to let Bob know the information if he pays her \$100. Bob knows the general nature of the information (but not its details) and, if he could actualy see it, he would immediately recognize that it is indeed valuable (and certainly worth \$100 to him). But Bob has only \$5, and so he suggests to Alice that, in return for paying her \$5, they engage in a protocol in which he has a 5% chance of getting the information. Design a protocol that achieves this and is such that (i) no cheating by Bob is possible, and (ii) after 3 hours have passed Alice can prove to Bob that she did not cheat when the protocol was performed 3 hours earlier.

Consider the following protocol for logging into Carol's machine (we describe it assuming that Carol already knows that Alice wants to login). It uses single-key crypto, and E_X means the same thing as in wide-mouth frog (i.e., encryption with the secret key that X shares with T).

1. $C \xrightarrow{T_C, R_C} A$
   where T_C is a timestamp and R_C is a random number generated by C.
2. $A \xrightarrow{LoginRequest, A, E_A(T_A, R_C, H(passwd), C)} T$
   where *LoginRequest* is just an indication that the purpose is to login, and *passwd* is A's password for her account with C.
3. After verifying the freshness of the timestamp, Trent does:
   $T \xrightarrow{LoginMsg, E_C(T_T, A, H(R_C, H(passwd)))} C$
   where T_T is a timestamp and $LoginMsg$ is an indication that this is for login.
4. C verifies freshness of the timestamp T_T, and correctness of R_C and $H(passwd)$.

Assume an environment in which both this login protocol (which we call LOGIN), and the wide-mouth frog (WMF) protocol, take place (both using the same key shared with Trent, i.e., the same E_A, E_B, E_C for both protocols). Show how, if Carol is in fact Mallory, she can impersonate Alice with Bob (i.e., can convince Bob that he is Alice).

Alice proves her presence to Bob's machine through the following protocol (in which all encryptions are single-key, and T is the trusted Trent).

1. $A \xrightarrow{A} B$.
2. $A \xleftarrow{R_B} B$
   where R_B is a random number generated by B; from here on B "remembers" the association between A and R_B (at least for a while — there is a "timeout" mechanism so that B "forgets" the association if the next 3 steps are not all completed within a certain amount of time).
3. $A \xrightarrow{E_{AT}(R_B)} B$
   where $E_{AT}(\cdot)$ denotes encryption with the key shared by A and T.
4. $B \xrightarrow{E_{BT}(A, E_{AT}(R_B))} T$.
5. $T \xrightarrow{E_{BT}(R_B)} B$
   and B, after retrieving R_B, allows the user associated with R_B (in this case A) access to his machine.

Just like Alice, Carol and others (but not Mallory) also have accounts on Bob's machine, and use a similar protocol to log on.

Show how Mallory could illegally gain access to Alice's account on Bob's machine (give the full details of how this is done).

Hints. Mallory tries to log on as "Alice" at about the same time as Carol is trying to log on. The attack would not be possible if, in Step 5, $E_{BT}(R_B)$ were replaced by $E_{BT}(A, R_B)$.

Suppose that the protocol is modified so that Step 5 now consists of:

$$T \xrightarrow{E_{BT}(A, R_B)} B.$$

Also assume that Mallory has a machine to which Alice logs on using the same protocol. Show how Mallory could illegally gain access to Alice's account on Bob's machine (give the full details of how this is done).

Hints. Mallory tries to log on as "Alice" at the same time as Alice is trying to log on to Mallory's machine. The attack would not be possible if, in steps 3 and 4 of the protocol, $E_{AT}(R_B)$ were replaced by $E_{AT}(B, R_B)$.

The following protocol for mutual authentication (using public key crypto) between Alice and Bob assumes that *they already have each other's key certificates* (from a trusted certificate authority).

1. $A \overset{E_B(A, R_A)}{\longrightarrow} B$
   where R_A is a random number generated by A and $E_B(\cdot)$ denotes encryption with Bob's public key.

2. $A \overset{E_A(R_A, R_B)}{\longleftarrow} B$
   where R_B is a random number generated by B.

3. $A \overset{E_B(R_B)}{\longrightarrow} B$.

Show how Mallory could trick Bob into thinking he is Alice (give the full details of how this is done).
Hint. Mallory does so as Alice engages in the protocol with him (i.e., he is Mallory to her).

Alice has an item x, and Bob has a set of five distinct items $y_1, \ldots, y_5$. Design a protocol through which Alice (but not Bob) finds out whether her x equals any of Bob's five items; Alice should not find out anything other than the answer ("Yes" or "No") to the above question, and Bob should not know that answer. Do not use a hash-based solution because even though the probability of a collision is small, Alice requires that no such collision can occur (but using encryption is fine, because in that case two distinct items that are encrypted with the same key will result in two different ciphertexts).

Alice has a random-bit generator that is defective in that it is *biased*: It generates more 1's than 0's (but the bits it generates are independent of one another, and are identically distributed). That is, if p_0 denotes the probability that the next bit generated is 0, and p_1 denotes the probability that the next bit generated is 1, then $p_1 > p_0$ (of course $p_0 + p_1 = 1$). However, what Alice really needs is a random-bit generator that is *unbiased*, i.e., one for which $p_0 = p_1 = 0.5$. Explain how Alice's biased random-bit generator can be used to generate an unbiased random bit string; your scheme should not assume that Alice knows p_0 or p_1 (she just knows that $p_1 > p_0$), and should not use hash functions or encryption.

A bank B allows its customers to withdraw cash from their accounts at hundreds of specialized automated teller mahines (ATMs) that are only for cash withdrawals (not for checking balances or performing other transactions). The ATMs operate in the following way. (In what follows E_B refers to encryption with the bank's secret key, in a single-key cryptosystem.) The bank asks the customer C to select a secret number (called "personal identification number", denoted by $PIN(C)$). Then the bank issues the customer C a special magnetized card that contains the following two pieces of information (on separate portions of the magnetized strip on the card):

- The customer's account number at the bank (call it $AcNr(C)$).
- $E_B(PIN(C))$.

Each ATM of that bank can perform an $E_B(\cdot)$ computation, and also stores a list of all the valid account numbers. It does not store the dollar balance in each account (each ATM limits cash withdrawals to no more than \$200 per day for each account, and each account contains at least \$500 — the bank automatically closes an account whose balance falls below the \$500 minimum). When the customer C wants to withdraw cash from an ATM, C inserts the card and the ATM reads the information on it and then challenges C to enter $PIN(C)$. The ATM then (i) verifies that the $AcNr(C)$ that it read from the card is on its list of valid account numbers, and then (ii) encrypts (i.e., does an $E_B(\cdot)$) what C just entered and verifies that the result equals the $E_B(PIN(C))$ that is stored in the card. If both (i) and (ii) are successfully verified the ATM allows the customer to withdraw the cash (subject to the constraint that the total amount withdrawn by C that day from that ATM does not exceed \$200). The ATM also stores a record of the transaction that consists of the account number and the amount just withdrawn. At midnight every day all the ATMs communicate with the bank's main computer, which updates all the customer accounts by subtracting from their balances the amounts of cash withdrawn that day. This *off-line* operation of the ATM allows the customers to quickly withdraw cash even when the network is down or very slow (at peak-hours during the day); contrast this to *on-line* operation, which would have required communication with the bank's main computer before a transaction can complete (and would have been problematic if the network was down or very slow at the time of the transaction).

Note that, if the card is stolen from the customer, the thief cannot obtain $PIN(C)$ from the card because it is encrypted (this is why it is $E_B(PIN(C))$ rather than $PIN(C)$ that is stored on the magnetic strip of the card — the latter would be insecure because *the information on the magnetic strip of a card is easy to read and modify*).

1. Show how a dishonest customer C' who knows $AcNr(C)$ can steal from C (by withdrawing cash from the account of C).

2. Suppose the theft occurs as in the above question 1. When C receives the end-of-month statement from the bank, C realizes what happened and complains to the bank. Can the bank tell for sure whether the theft really occurred, i.e., that C is not lying ?

3. Without doing any modification to the card itself, explain how you would modify the ATM protocol so that the identity of C' is revealed if C' were to steal from C as in question 1. That is, when C gets the end-of-month statement and complains about illegal withdrawals from her account, the bank should at least be able to find out which C' committed the crime.

4. In this question you are allowed to modify both the card's design and the ATM protocol: What should the card contain to prevent the theft by C' ? How would the corresponding ATM protocol be different ?

5. Explain how a dishonest customer C can withdraw much more than the \$200 daily limit from her own account. What would the bank have to do to make this impossible?