A-4 What is the difference between a three-party scheme and a four-party scheme for credit card payments?

In a four-party scheme the acquirer(dealing with the merchant) and the issuer(dealing with the customer) are two separate entities. This comes with a cost as there is a interchange fee between the acquirer and the issuer. Having different entities for the issuer and acquirer opens up the possibility for other institutions to issue to have a partnership with a bank and issue their own card, e.g. CircleK cards and such. Visa and Mastercard use a four-party scheme.

In a three-party scheme the issuer and the acquirer is the same entity and you are able to bypass the interchange fee between an issuer and an acquirer. This limits the competition to just compete against other brands. An example of a three-party scheme is American Express.

A-5 How does the Merchant verify the dual signature in SET?

The merchant needs the payment information message digest (PIMD) and the order information(OI). This information is sent together with the cardholders certificate, the dual signature and the encrypted payment information in the purchase request message. The merchant can then verify the dual signature by hashing the order information(OI) and then concatenating this result together with the received payment information message digest(PIMD). This concatenated string is then hashed again and checked against the decrypted dual signature(decrypted using the customers public key which is contained in the certificate). If these two are equal, the dual signature is verified.

A-6 In SET, why is the Payment Information first symmetrically encrypted and not immediately encrypted with the Gateway's public key?

Because by signing with own symmetric key it prevents the user of denying of having taking some action. It's also a way to prove that only the real owner has used his/her own payment information. This encryption step provides non-repudiation.

A-17 How can the cut–and–choose technique be used to make sure that identifying information is properly added into an untraceable coin?

Each untraceable coin is linked with an identifier which can identify a user in case of double spending. When a user wants to withdraw coins from the bank each coin is represented by $2k$ quadruples of random numbers (mod n), chosen by the user; a_i, c_i, d_i, r_i. The user then computes $2k$ values

$$B_i = r_i^3 f(x_i, y_i) \bmod n$$

where

$$x_i = h(a_i, c_i) \ , \ y_i = h(a_i \oplus ID, d_i)$$

All these B_i values are sent to the bank. As seen in the equation for y_i the ID for the user is embedded into y_i.

The bank then randomly chooses k indices out of the $2k$ that was sent and sends these indices back to the user. For each index the user has to respond with the quadruple a_i, c_i, d_i, r_i. The bank can then verify the quadruple by computing B_i, knowing the ID for the user. If the user has included his/her identification in all of these indices then with high probability(not guaranteed) it is included in the remaining k B_i.

A-20 How is Alice's identity revealed if she double spends a coin in the untraceable E-cash scheme?

When Alice buys something from Bob she sends the signature S of the coin, which she computed using k quadruples a_i, $a_i \oplus ID$, c_i, d_i. In order to verify the coin Bob then generates a random binary vector of length k, $z = (z_1, \ldots, z_3)$, and sends this back to Alice. Alice then uses all the indices not used when generating the signature and orders them from 1 to k when responding to the binary vector by the following rules:

- If $z_j = 0$, then Alice sends x_j, $a_j \oplus ID$, d_j to Bob
- If $z_j = 1$, then Alice sends y_j, a_j, c_j to Bob

This allows Bob to compute $f(x_j, y_j)$ for all indices j and verify the signature. Note that Bob can not compute the ID since he does not have both a_j and $a_j \oplus ID$ for any j.

Bob can at anytime send the signature S, the random binary vector z and Alice's response to the bank. The bank then verifies the signature S and credits Bob's account.
Since the bank receives the random binary vector z and Alice's response it now has either a_j or $a_j \oplus ID$ for each index j. If Alice decide to double spend that coin with another merchant, this merchant will also create a random binary vector z' and Alice's response to that vector will then be forwarded to the bank when the new merchant wants to be credited money for the purchase. If there exists a index j such that $z_j \neq z'_j$ then the bank will have both a_j and $a_j \oplus ID$ and can then ID can simply be extracted by computing $a_j \oplus a_j \oplus ID = ID$

A-22 Briefly explain the differences between session-level aggregation, aggregation by intermediation and universal aggregation.

All three aggregation works by combining several micropayments into a larger macropayment, to avoid relatively large transaction fees, before any deposit and withdrawals are done on the actual bank accounts(hence the name, aggregation).

Session-level aggregation only works between one customer and one merchant. The user makes several micropayments to the merchant and at the end of some time interval all these micropayments are collected and added together to one macropayment which the user then get debited for. This is a one-to-one relationship.

Universal aggregation takes the limitation that one merchant only can collect micropayments from one user away and changes this to collect micropayments from many users and aggregate these micropayments into one macropayment. This therefore changes the relationship to a many-to-many variant.

Aggregation by intermediation works by adding a third party that keeps track of all micropayments. The third party works like a database between the users and the merchants. When enough micropayments have been made these are aggregated into one large macropayment using the regular banks. A drawback of this is of course the additional party that is involved and wants to make money out of this.

A-28 Compare the PayWord protocol and the Peppercoin-like protocol in the lecture notes from the point of view of the customers, both in terms of what they pay, and in terms of what they need to compute to make a purchase.

In the PayWord protocol the user pays exactly the amount he/she "owes". To guarantee this a hash chain is used. When a user, C, wants to make a purchase from a new merchant, M, the user computes a hash chain of n hashes: $w_i = h(w_{i+1})$ where $i = n \ldots 0$. The user then sends a commitment $S = \{M, W_0, C\}$ to the merchant.When the user wants to buy something that costs 1 price unit he/she sends a PayWord $P = (w_i, i)$ to the merchant and for every new item(or currency unit) i is incremented. If something costs m units then i is incremented by m. At the end of the day the merchant can send the PayWord with the largest i together with the commitment S to the bank and the bank can verify the PayWord by hashing it i times and checking it against the commitment. If all checks out the bank can credit the merchant and debit the user.

In the Peppercoin-like protocol the user is guaranteed to not pay more than he/she owes but can also get away with paying nothing. Instead the psychological disadvantage of the variance in payments is moved to the banks. The Peppercoin-like protocol is built upon mathematical statistics and its quite simple from the users perspective. When the user makes a purchase he/she sends $\{T, S\}_{PRI_U}$ to the merchant, where T is all the information about the current purchase and S is a serial number which is incremented for every purchase. The purchase results in a macropayment if $F(\{\{T, S\}_{PRI_U}\}_{PRI_M})$, where F is a function that maps a binary string to a real number between 0 and 1, is less than some quote between the micropayment and the macropayment amount. This macropayment is payed by the bank to the merchant and the user is debited the amount he/she owes. In the long run this will even out for the banks.

A-29 What is meant by a probabilistic payment? How does the Electronic Lottery Tickets scheme differ from Peppercoin from the user's perspective? How do they differ from the Merchant's perspective?

A probabilistic payment is a payment that is done with a certain propability. The math behind the probability that a payment should be done differs from protocol to protocol.

From the users perspective the Peppercoin-scheme is very much more inviting as the user does not risk to pay alot more than he purchased for. This variance in the payments is instead moved to the banks, unlike the Electronic Lottery Tickets scheme where the user has this psychological disadvantage.

From the merchants perspective it is about the same. The merchant gets payed a macropayment once a probability condition is met. The probability condition in the Electronic Lottery Scheme is $m_i \bmod \gamma/\mu = w_i \bmod \gamma/\mu$ where m_i and w_i are the merchants and users calculated hash chains, respectively.
In the Peppercoin scheme the probability is instead calculated by some function F that maps a binary string to a real number between 0 and 1. If that number is less than the probability γ/μ a macropayment is done by the bank to the merchant and the user is debited the amount that he owes.
The difference here is that the merchant doesn't have to keep track of the users individual hash chains.