1ai) When 2 miners mine a block simultaneously, the rejected ones are the one with less proof of work or is propagated less. The rejected blocks are known as uncle blocks. In bitcoin this blocks are known as orphan blocks. Unlike bitcoin orphan blocks, uncle blocks are incentivized, several reasons include decreasing the centralization of incentives of large mining pools and increasing security of the network by supplementing the work on the blockchain.

Aii) The uncle blocks are purposefully incorporated into Ethereum's consensus method by a process called "GHOST: Greedy Heaviest Object Sub Tree. nodes will get the number of uncles mined for the last seven blocks in each subtree. That number is, in addition to the number of blocks in that subtree, used to calculate the tree’s weight; the heaviest tree is then said to be the 'correct' one. This particular use increases the security

Iii) Markov decision process allows one to know the optimal adversarial strategies by comparing the performance and security of blockchain systems with different parameters. This allows one to see how likely is someone going to do honest mining compared to un-honest mining in a particular scenario

Bi)10 minutes for bitcoin, 15 seconds for ETH

Ii) A faster block generation interval would mean more stale blocks because more people are generating block this reduces security, whereas a slower block generation would increase security. On the other hand, faster block generation means more txn can be put on the blockchain which is better for scalability reasons. Thus there is a balance btw security and scalability, fast block generation rate would increase scalability but reduce security.

Ci) UTXO means unspent transaction output. The sum of output must be less or equal than sum of input. If the output is less, the difference is transaction fees. Diagram in slides. Output one guys to vendor another goes back to your account.

Ii) Growing UTXO set requires an increase in storage (although currently it is rather small, 4-5GB). Not a huge problem for scalability given the prevalence of storage today and also if UTXOs are regularly consolidated then even less so.

Reduction in block validation time, as transactions can be parallelised for validation. (maybe???)

2i) Coinjoin requires multiple parties to jointly sign an agreement to mix their coins in a Bitcoin transaction where the output of the transaction leaves the participants with the same number of coins but the addresses have been mixed to make external tracking difficult.

Ii) Because one wont know who is paying who using coinjoin since all the transaction are mixed.

Iii) It has a small anonymity set, it is restricted to the number of participants. There is also a need for a trusted 3rd party to mix the transactions, 3rd party might be able to see who pays who

Iv) I am just gonna follow a concept similar to mixing pools. So people put coins into a pool. And then provides the vendor a smart contract (a zero knowledge proof) that he/she owns a certain amount of coin in the pool.

Bi) Hard fork -- since nodes that do not update will view the new blocks as too big and so invalid

Ii)Softfork, cause without chainID it would still work? Old nodes could still validate blocks and transactions (the chainID didn’t break the rules), but they just wouldn’t understand them.

Iii)Hardfork since change of definition of OPverify, old version no longer accepted

3a i)Anyone because private for the smart contract means they can't be accessed or modified from other smart contracts. But their values can be read freely outside the blockchain by anyone

b i) re-entry attack is possible because balance is only updated after the call function. This is possible because the call function makes an external call to another untrusted contract before it resolves any effects. If the attacker can control the untrusted contract, they can make a recursive call back to the original function, repeating interactions that would have otherwise not run after the effects were resolved.

c i)use the transfer function instead because the transfer function is limited by a gas limit (2300). Also, reverts the transaction execution if an error is thrown

N.B. transfer and send have since been deprecated: <https://consensys.github.io/smart-contract-best-practices/recommendations/#dont-use-transfer-or-send>

ii) We get overflow so the result is 0

4ai) In slides, pick a few, eg HTLC, Time-locks, Plasma...

Ii) The bitcoin scalability problem is the limited rate at which the bitcoin network can process transactions. The on-chain transaction processing capacity of the bitcoin network is limited by the average block creation time of 10 minutes and the block size limit of 1 megabyte.

B

i) Payment channels are first created by one party A sending part of a bitcoin to another party B, and then part B sending part of that remainder back to party A. This is done off chain. The two parties agree on a shared state of the blockchain (anchor transaction)

ii) Throughput is increased as it is done off-chain and therefore doesn’t require the usual necessary committal to the primary blockchain. As most of the intermediate transactions are not done on the blockchain but through the payment channel.

iii) HTLC and something else? (Multisignasture?)

iv) Spillman, Duplex micgropayment, lightning

V) assume a party doesn’t collude with a miner for the final transaction on the chain

* The blockchain is functioning well (confirming transactions quickly)
* Channel nodes can keep secret data safe
* There aren't any significant bugs in the software

Vi) Require all parties along the payment route to be online simultaneously

* Payment amount is limited by capacity of channels along the payment route
* May tend toward “hub” model with less decentralisation (easier to set up channel with a well-connected node, like an exchange etc)
* May take a while to settle in the case of dispute

C) commit chains are when off-chain transactions are periodically updated and validated on the blockchain. So not all transactions are on the main blockchain, but instead every X transactions the state is recorded on the blockchain.

Commit chains can involve many parties and compared to payments channels are ongoing states. Users can join and transact on the commit-chain with the ability to withdraw to the main blockchain whenever they want.