MONASH University

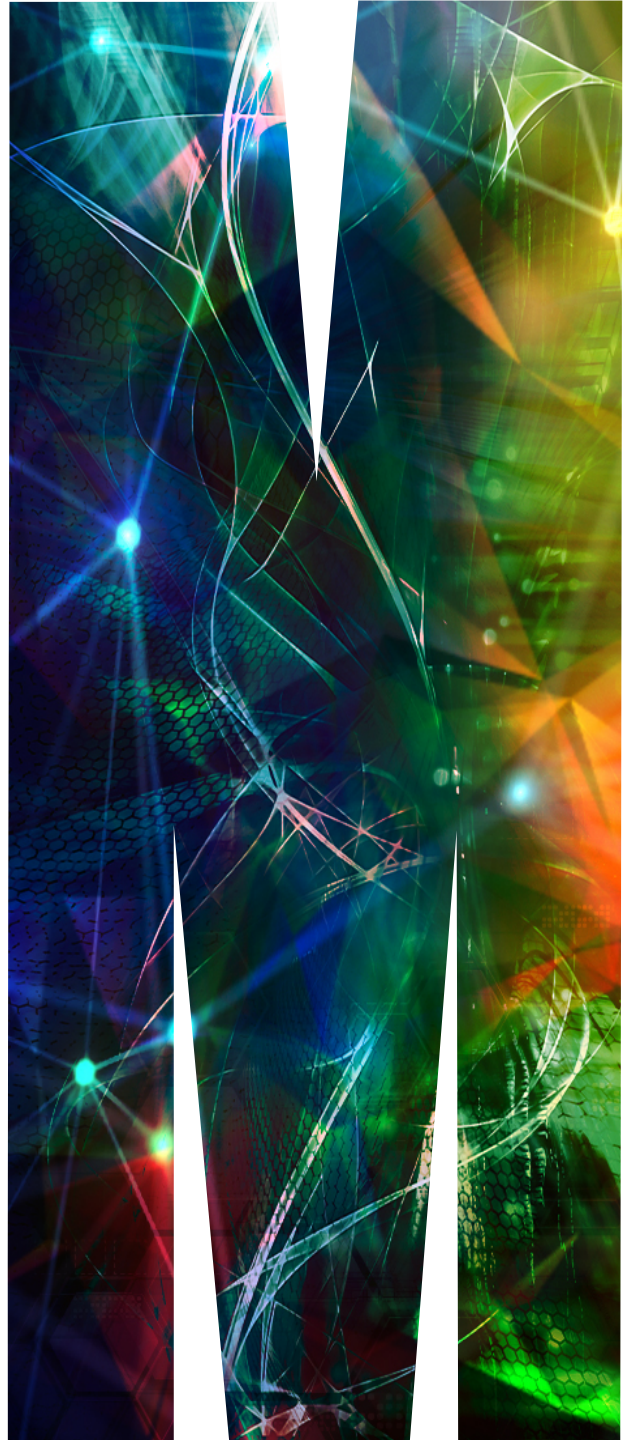
FIT5214: Blockchain

Lecture 7: Proof-of-Stake (PoS)

Lecturer: Rafael Dowsley

rafael.dowsley@monash.edu

https://dowsley.net



Unit Structure

- Lecture 1: Introduction to Blockchain
- Lecture 2: Bitcoin
- Lecture 3: Ethereum and Smart Contracts
- Lecture 4: Proof-of-Work (PoW)
- Lecture 5: Attacks on Blockchains
- Lecture 6: Class Test/Alternatives to PoW
- Lecture 7: Proof-of-Stake (PoS)
- Lecture 8: Privacy
- Lecture 9: Byzantine Agreement
- Lecture 10: Blockchain Network
- Lecture 11: Payment Channels
- Lecture 12: Guest Lecture

Unit Structure

- Lecture 1: Introduction to Blockchain
- Lecture 2: Bitcoin
- Lecture 3: Ethereum and Smart Contracts
- Lecture 4: Proof-of-Work (PoW)
- Lecture 5: Attacks on Blockchains
- Lecture 6: Class Test/Alternatives to PoW
- Lecture 7: Proof-of-Stake (PoS)
- Lecture 8: Privacy
- Lecture 9: Byzantine Agreement
- Lecture 10: Blockchain Network
- Lecture 11: Payment Channels
- Lecture 12: Guest Lecture

Some Problems of PoW

- Control of the network is very centralised in a few mining pools.
- A huge amount of energy is wasted computing useless hash outputs.
- Migration of hash power to do 51% attacks in platforms with smaller amounts of total hash power.

Good Properties of PoW

As proved in “The Bitcoin Backbone Protocol: Analysis and Applications” (Eurocrypt 2015), a system like Bitcoin achieves the following interesting properties:

- Common Prefix: Honest miners share a consistent common prefix.
- Chain Growth: The number of blocks increases over time.
- Chain Quality: A guaranteed fraction of honestly contributed blocks.

These properties imply the persistence of the transactions in the ledger (after they are deep enough in the blockchain) and the liveness of the ledger.

Good Properties of PoW

As proved in “The Bitcoin Backbone Protocol: Analysis and Applications” (Eurocrypt 2015), a system like Bitcoin achieves the following interesting properties:

- Common Prefix: Honest miners share a consistent common prefix.
- Chain Growth: The number of blocks increases over time.
- Chain Quality: A guaranteed fraction of honestly contributed blocks.

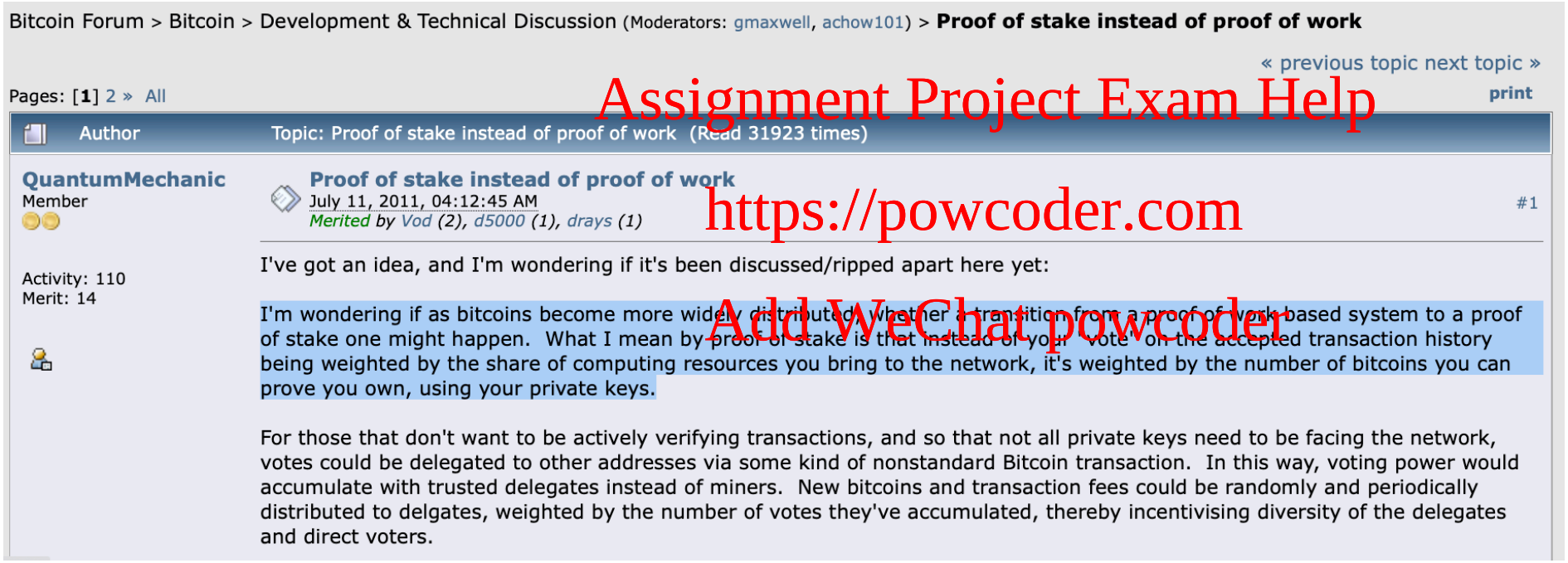
These properties imply the persistence of the transactions in the ledger (after they are deep enough in the blockchain) and the liveness of the ledger.

Moreover, it supports full dynamic availability: nodes can join and leave at will.

We want to keep those nice properties in a PoS solution.

Proof-of-Stake (PoS)

The concept was first proposed in a Bitcoin community forum on 11 July 2011.



Bitcoin Forum > Bitcoin > Development & Technical Discussion (Moderators: gmaxwell, achow101) > **Proof of stake instead of proof of work**

« previous topic next topic »

Pages: [**1**] 2 » All

print

Author	Topic: Proof of stake instead of proof of work (Read 31923 times)

QuantumMechanic
Member

Activity: 110
Merit: 14

Proof of stake instead of proof of work
July 11, 2011, 04:12:45 AM
Merited by Vod (2), d5000 (1), drays (1)

#1

I've got an idea, and I'm wondering if it's been discussed/ripped apart here yet:

I'm wondering if as bitcoins become more widely distributed whether a transition from a proof of work based system to a proof of stake one might happen. What I mean by proof of stake is that instead of your "vote" on the accepted transaction history being weighted by the share of computing resources you bring to the network, it's weighted by the number of bitcoins you can prove you own, using your private keys.

For those that don't want to be actively verifying transactions, and so that not all private keys need to be facing the network, votes could be delegated to other addresses via some kind of nonstandard Bitcoin transaction. In this way, voting power would accumulate with trusted delegates instead of miners. New bitcoins and transaction fees could be randomly and periodically distributed to delgates, weighted by the number of votes they've accumulated, thereby incentivising diversity of the delegates and direct voters.

https://bitcointalk.org/index.php?topic=27787.0

Proof-of-Stake (PoS)

PoW: vote by comparing power — more computing power, more votes
PoS: vote by number of coins — more coins, more votes

So, with **PoW**, the one with more than **50% computing power** controls the system
with **PoS**, the one with more than **50% coins** controls the system

Proof-of-Stake (PoS)

Original proposal:

1. Each vote is weighted by the number of owned Bitcoins
2. The proof of ownership is done by creating a signature using BTC private key
3. Users can delegate their votes to others
4. The winner gets some rewards

Group discussion:

Are there problems/ challenges?

Proof-of-Stake (PoS)

Original proposal:

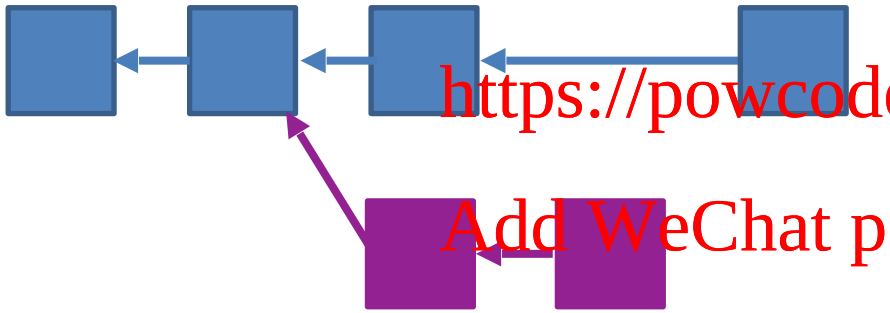
1. Each vote is weighted by the number of owned Bitcoins
2. The proof of ownership is done by creating a signature using BTC private key
3. Users can delegate their votes to others
4. The winner gets some rewards

Potential problems/challenges:

1. Users may not be always online for voting;
2. How to choose a delegator?
3. Too many users to cast the votes;
4. No cost in voting

Nothing at Stake Attack

No computing/storage resources are required to create blocks: several transaction histories can be easily generated by an adversary.



Nothing at Stake Attack

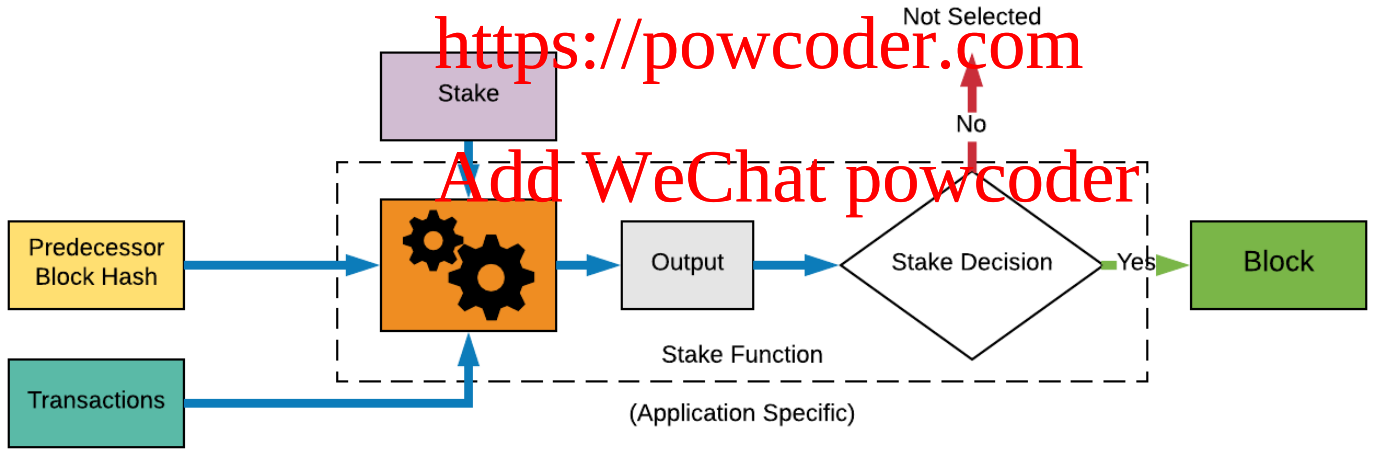
No computing/storage resources are required to create blocks: several transaction histories can be easily generated by an adversary.

An attacker that is selected as leader to add a block in the next slot, can add one block to each branch in order to guarantee his reward.

Proof-of-Stake (PoS)

There are different types of PoS, depending on the "things" put at stake:

- Proof-of-Lock/Deposit
- Balance-based
- …



Nakamoto-style consensus: e.g., Ouroboros
BFT-style consensus: e.g., Algorand

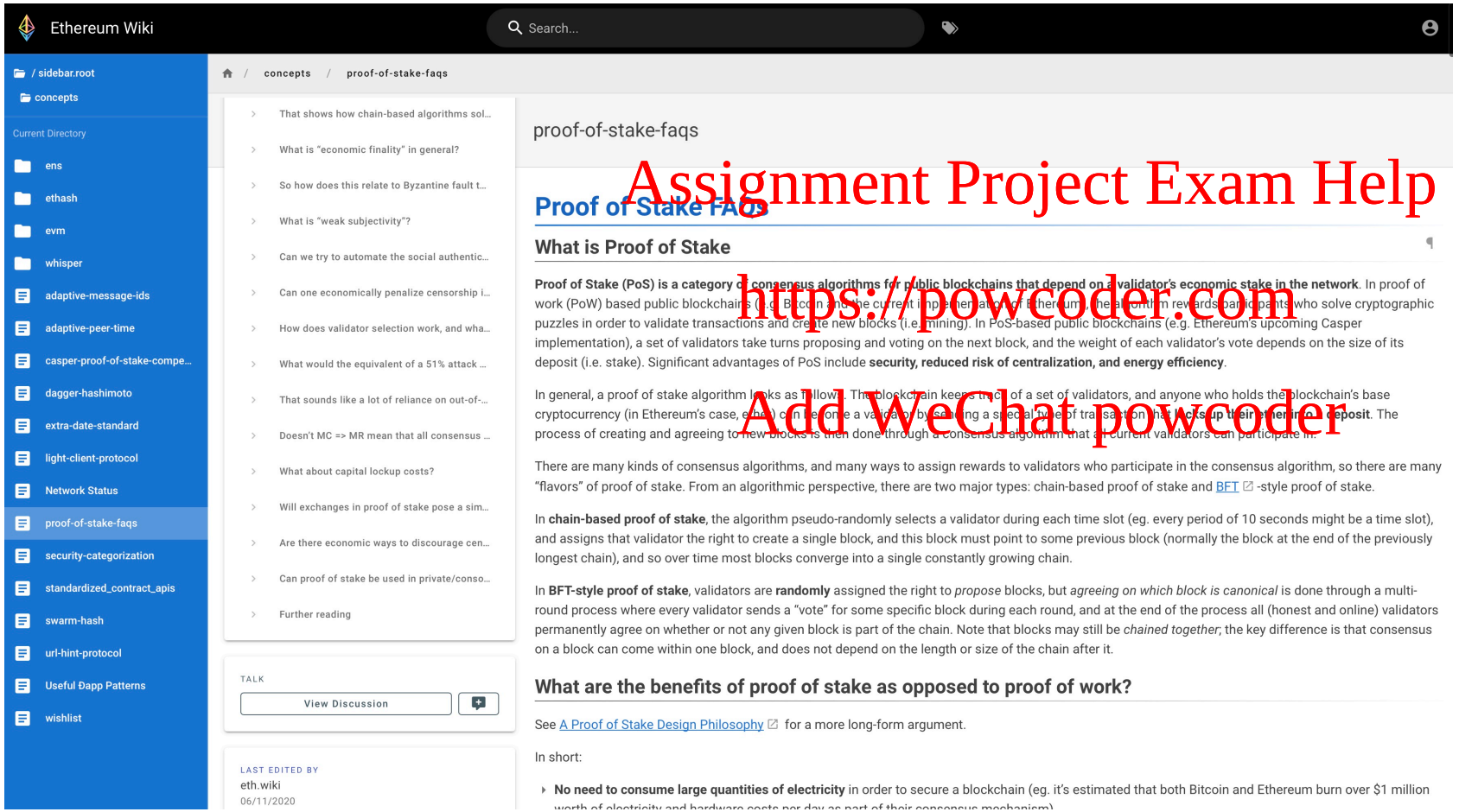
Proof-of-Lock/Deposit

1. **Each user votes by locking/depositing some coins**
2. Each vote is weighted by the number of locked/deposited coins
3. The prove of ownership is done by creating a signature using private key
4. The winner gets some rewards
5. **If misbehaved, the locked/deposited coins are destroyed**

This solves the Nothing-at-stake attack, as the attacker will be punished.

Proof-of-Lock/Deposit

Example: Ethereum Casper



https://eth.wiki/en/concepts/proof-of-stake-faqs

Discouragement Attacks

Consider the following scenario:

1. Users put deposit at stake
2. A set of N users is selected as validators
3. One of the N users is chosen to propose the block
4. If a block is validated by most (>N/2) validators,
   then block is validated, and the user gets reward;
   else, the user loses the deposit

Discouragement Attacks

Attacking strategy:

1. Put a lot of money to control >N/2 validators
2. Only accept its own blocks
    1. other honest validators lose deposit
    2. the attacker's profit does not change
3. Honest validators will leave the group to avoid losing money
4. All validators left are controlled by the attacker

Result:

In a long run, the attacker can use less money to gain all rewards, and gets full control of the blockchain.

Balance-Based PoS (Nakamoto-style)

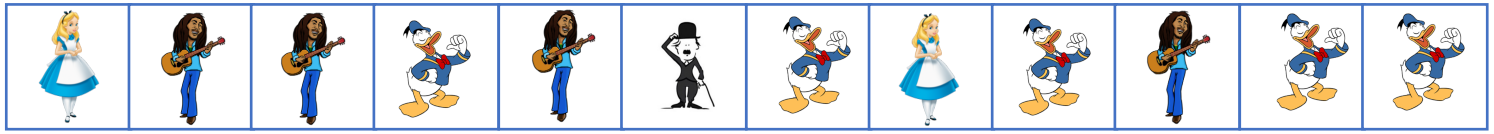
Nakamoto-style consensus dividing the time into slots. For each slot, select one leader to create the block corresponding to that slot.

Lottery: the slot leader is randomly selected with the users probabilities of being selected proportional to their stake.

Balance-Based PoS (Nakamoto-style)

Nakamoto-style consensus dividing the time into slots. For each slot, select one leader to create the block corresponding to that slot.

Lottery: the slot leader is randomly selected with the users probabilities of being selected proportional to their stake.

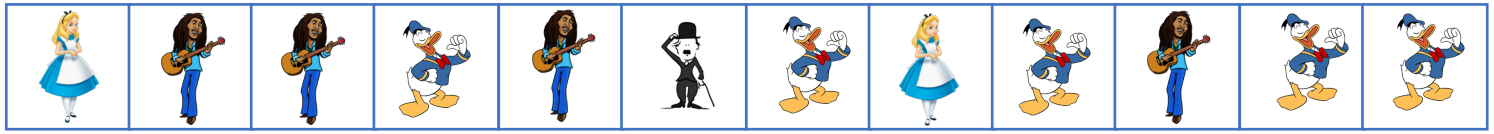


Assume trusted lottery for now.

Security Assumption

At all times the adversary controls less than 50% of the coins.

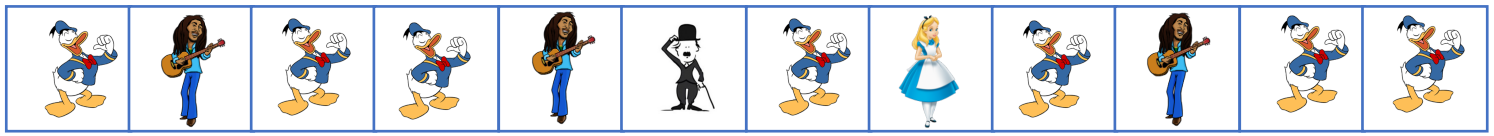
Is that enough?



Assume trusted lottery for now.

Adaptive Adversary

If the slot leaders are publicly known some time in advance and the adversary is adaptive, he can use that information to strategically corrupt parties with larger influence than expected.

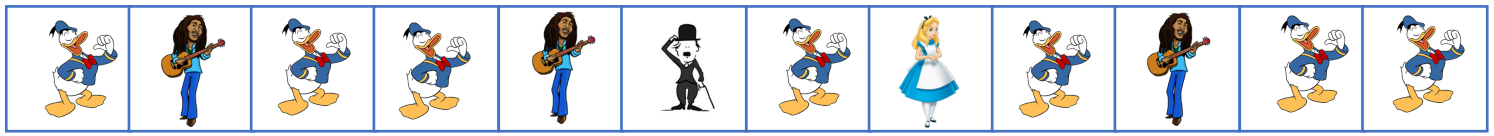


Assume trusted lottery for now.

Adaptive Adversary

If the slot leaders are publicly known some time in advance and the adversary is adaptive, he can use that information to strategically corrupt parties with larger influence than expected.

"Donald was very lucky in the lottery, I will corrupt him to lead more than half of the slots while only controlling 40% of the stake".



Assume trusted lottery for now.

MONASH University

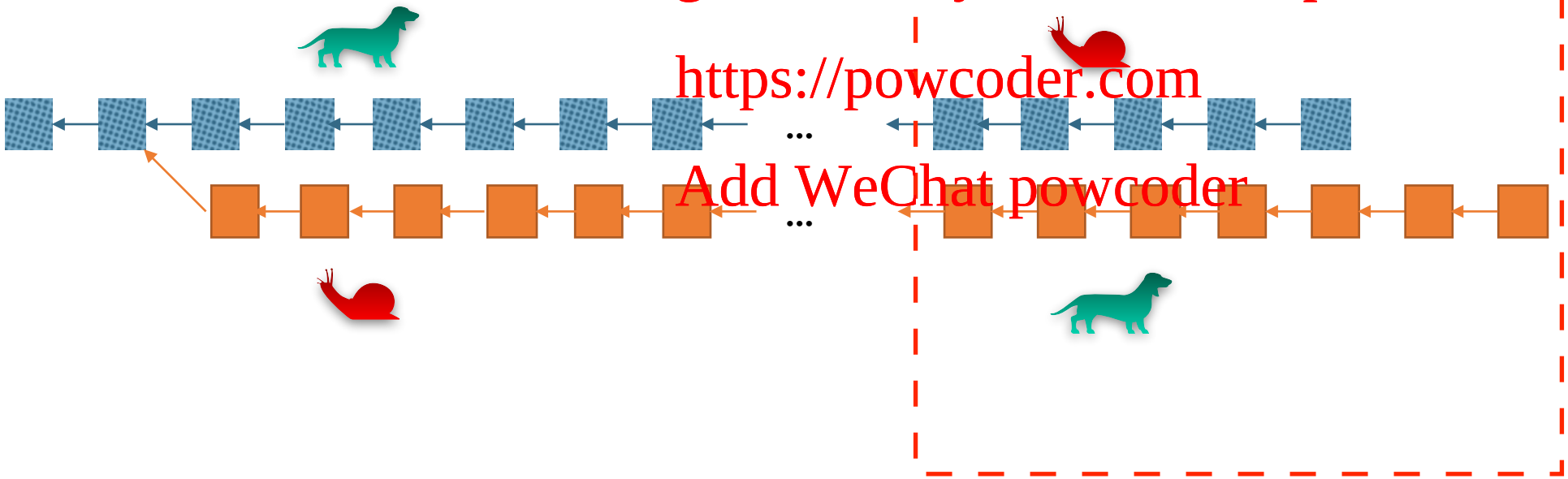
Long Range Attacks

Assuming the longest chain rule, the attacker only creates blocks in its side chain, but does not work on the honest main chain. This delays the growth of the main chain.

At beginning, the attacker is the minority, and creates blocks in the side chain slower than the honest blockchain

Long Range Attacks

Assuming the longest chain rule, the attacker only creates blocks in its side chain, but does not work on the honest main chain. This delays the growth of the main chain.



In the side chain, the attacker will **eventually** become a super majority as it is the only party gaining mining rewards, and can build the chain faster than the main chain. (It takes a long time.)

Posterior Attack

Variant of long range attack in which the attacker tries to obtain the secret keys of users that do not have any stake anymore, but used to control some stake in the past (i.e., they had voting rights in the past).

This helps the attacker to speed up the growth of the private side chain.

Ouroboros

Ouroboros PoS protocol was developed by IOHK to be used in the Cardano blockchain platform, the leading PoS blockchain platform nowadays.

☆	1	Bitcoin BTC Buy	$19,782.37	0.27%	0.45%	2.15%	$378,410,219,638	$31,695,154,831 1,601,668 BTC	19,147,625 BTC
☆	2	Ethereum ETH Buy	$1,638.00	0.48%	3.97%	6.44%	$200,080,782,814	$15,737,100,519 9,615,940 ETH	122,256,621 ETH
☆	3	Tether USDT Buy	$1.00	0.00%	0.00%	0.00%	$67,546,993,526	$44,127,394,681 44,424,545,217 USDT	67,545,704,104 USDT
☆	4	USD Coin USDC Buy	$1	0.03%	0.01%	0.00%	$51,843,230,792	$6,817,269,757 6,817,844,020 USDC	51,848,028,033 USDC
☆	5	BNB BNB Buy	$279.29	0.14%	0.27%	2.58%	$45,043,174,422	$811,182,218 2,905,522 BNB	161,337,261 BNB
☆	6	Binance USD BUSD	$1.00	0.03%	0.04%	0.02%	$19,433,827,406	$8,004,581,006 8,004,332,542 BUSD	19,433,224,173 BUSD
☆	7	Cardano ADA	$0.4963	1.01%	0.29%	10.62%	$16,934,607,586	$734,494,449 1,482,557,039 ADA	34,182,044,153 ADA
☆	8	XRP XRP Buy	$0.3335	0.27%	0.69%	1.14%	$16,543,841,465	$972,412,420 2,918,117,046 XRP	49,646,492,379 XRP

Other companies are implementing variants of Ouroboros in their blockchain products.

MONASH University

Ouroboros

- Ouroboros “Classic” (Crypto 2017): strong mathematical framework used to proof its security, semi-adaptive adversaries, synchronous protocol.

Ouroboros

- Ouroboros “Classic” (Crypto 2017): strong mathematical framework used to proof its security, semi-adaptive adversaries, synchronous protocol.
- Ouroboros Praos (Eurocrypt 2018): + only semi-synchronous, secure against adaptive adversaries.

Ouroboros

- Ouroboros “Classic” (Crypto 2017): strong mathematical framework used to proof its security, semi-adaptive adversaries, synchronous protocol.
- Ouroboros Praos (Eurocrypt 2018): + only semi-synchronous, secure against adaptive adversaries.
- Ouroboros Genesis (CCS 2018): + full dynamic availability (parties join and leave at will), bootstrapping from genesis block.

Ouroboros

- Ouroboros “Classic” (Crypto 2017): strong mathematical framework used to proof its security, semi-adaptive adversaries, synchronous protocol.
- Ouroboros Praos (Eurocrypt 2018): + only semi-synchronous, secure against adaptive adversaries.
- Ouroboros Genesis (CCS 2018): + full dynamic availability (parties join and leave at will), bootstrapping from genesis block.
- Ouroboros Chronos (Eurocrypt 2021): + relaxation of clock synchrony assumption, bootstrapping time from genesis block.

Ouroboros

- Ouroboros “Classic” (Crypto 2017): strong mathematical framework used to proof its security, semi-adaptive adversaries, synchronous protocol.
- Ouroboros Praos (Eurocrypt 2018): + only semi-synchronous, secure against adaptive adversaries.
- Ouroboros Genesis (CCS 2018): + full dynamic availability (parties join and leave at will), bootstrapping from genesis block.
- Ouroboros Chronos (Eurocrypt 2021): + relaxation of clock synchrony assumption, bootstrapping time from genesis block.
- Ouroboros Crypsinous (S&P 2019): privacy-preserving PoS.

Ouroboros

- Ouroboros “Classic” (Crypto 2017): strong mathematical framework used to proof its security, semi-adaptive adversaries, synchronous protocol.
- Ouroboros Praos (Eurocrypt 2018): + only semi-synchronous, secure against adaptive adversaries.
- Ouroboros Genesis (CCS 2018): + full dynamic availability (parties join and leave at will), bootstrapping from genesis block.
- Ouroboros Chronos (Eurocrypt 2021): + relaxation of clock synchrony assumption, bootstrapping time from genesis block.
- Ouroboros Crypsinous (S&P 2019): privacy-preserving PoS.

Here we focus on a high-level idea of Ouroboros Praos.

Communication Model

- **Semi-synchronous communication model**
- Participants have synchronised watches. Time divided into slots.
- Any message sent by an honest player is delivered to all honest players (diffusion network) within at most Δ slots.
- Adversary has full control over delays (within the Δ-bound).
- Δ is unknown to the protocol (security provably degrades gracefully with increasing Δ).

Adversarial Model

- Adversary can corrupt any party immediately.
- Adversary has minority stake at all times.
- Stake shifts at bounded rate.

Chain Selection Rule

- Adopt a valid new chain if it is longer and does not fork by more than k blocks from local chain.

Chain Selection Rule

- Adopt a valid new chain if it is longer and does not fork by more than k blocks from local chain.
- **Protects against long-range attacks.**

Chain Selection Rule

- Adopt a valid new chain if it is longer and does not fork by more than k blocks from local chain.
- **Protects against long-range attacks.**
- Simplified model: no newcomers, full participation.
- Ouroboros Genesis has a slightly more elaborate chain selection rule in order to provide full dynamic availability and support bootstrapping from the genesis block.

Stronger Crypto Tools

- Key-Evolving Signature Scheme (KES)
- Verifiable Random Function (VRF)

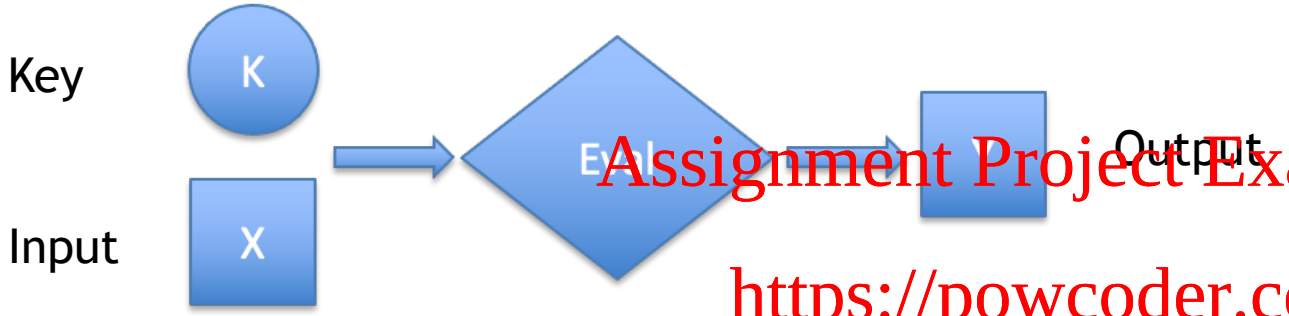
Key-Evolving Signature Scheme (KES)

- As normal signature scheme, KES should be unforgeable.
- Key update: after each use, the secret-key is updated and the old version of the secret-key is erased. The public-key remains the same.
- **Impossible to forge old signatures with new keys.**

Key-Evolving Signature Scheme (KES)

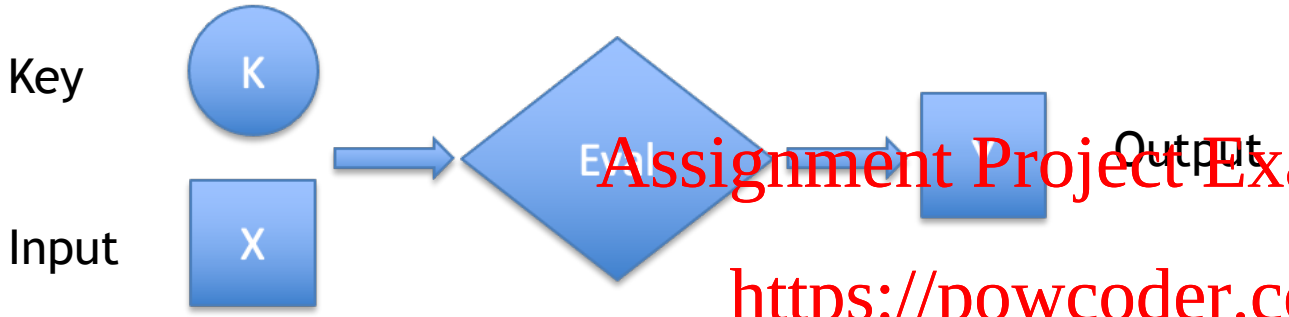
- As normal signature scheme, KES should be unforgeable.
- Key update: after each use, the secret-key is updated and the old version of the secret-key is erased. The public-key remains the same.
- **Impossible to forge old signatures with new keys.**
- **Used in Ouroboros for signing blocks.**
- **Purpose:** Protect previous actions and therefore help achieving security against adaptive adversaries.

Pseudorandom Function (PRF)



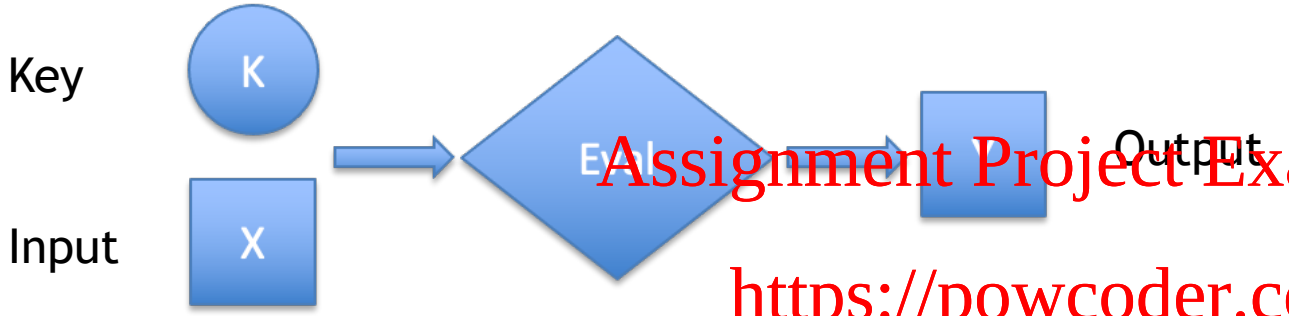
- Pseudorandomness (intuition): the output cannot be distinguished from a truly random value.

Pseudorandom Function (PRF)



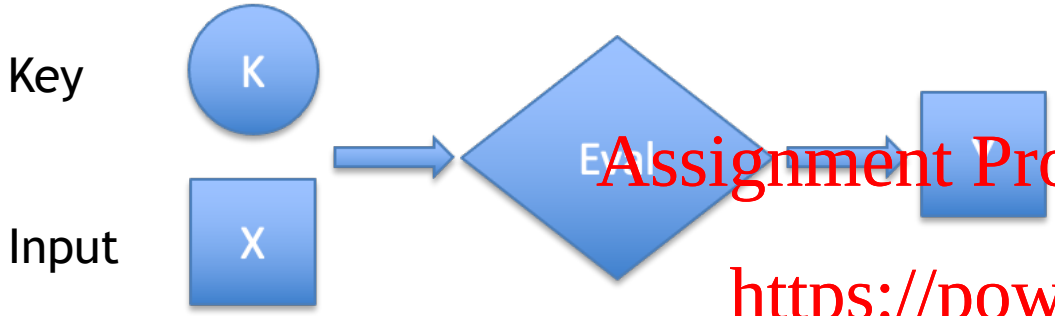
- Pseudorandomness (definition):
  - Adversary doesn't know the key.
  - Adversary have oracle access to Eval (he gives the input, gets output).

Pseudorandom Function (PRF)



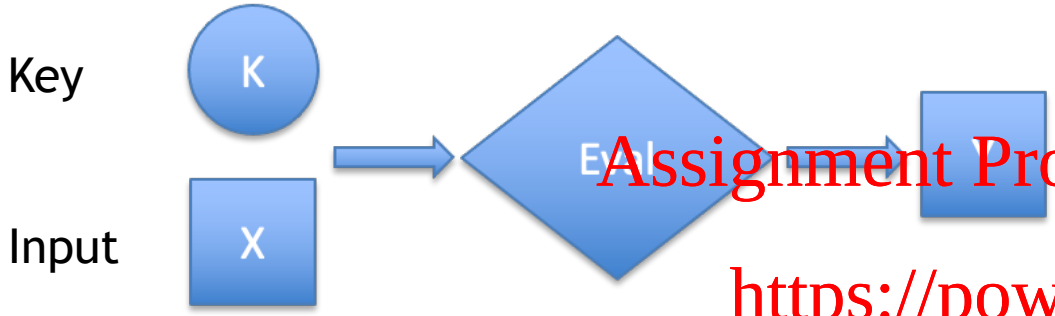
- Pseudorandomness (definition):
  - Adversary doesn't know the key.
  - Adversary have oracle access to Eval (he gives the input, gets output).
  - Adversary choose an input X to be challenged on (he cannot query the oracle on X).

Pseudorandom Function (PRF)



- Pseudorandomness (definition):
  - Adversary doesn't know the key.
  - Adversary have oracle access to Eval (he gives the input, gets output).
  - Adversary choose an input X to be challenged on (he cannot query the oracle on X).
  - With probability 1/2, he is given the result Y of evaluating the function on input X. With probability 1/2, he is given a uniformly random value.

Pseudorandom Function (PRF)



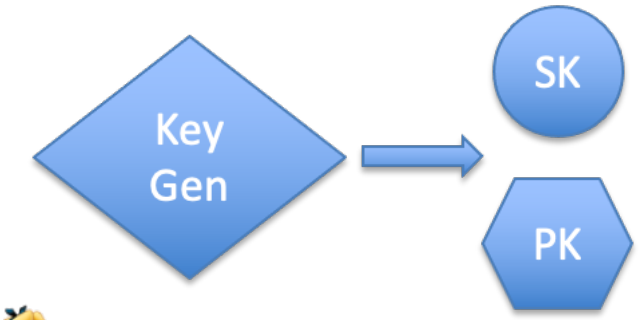
- Pseudorandomness (definition):
  - Adversary doesn't know the key.
  - Adversary have oracle access to Eval (he gives the input, gets output).
  - Adversary choose an input X to be challenged on (he cannot query the oracle on X).
  - With probability 1/2, he is given the result Y of evaluating the function on input X. With probability 1/2, he is given a uniformly random value.
  - A function is pseudorandom if no adversary can distinguish the two cases with probability non-negligibly better than 1/2.

Verifiable Random Function (VRF)

- High-level idea: A PRF that can be publicly verified.

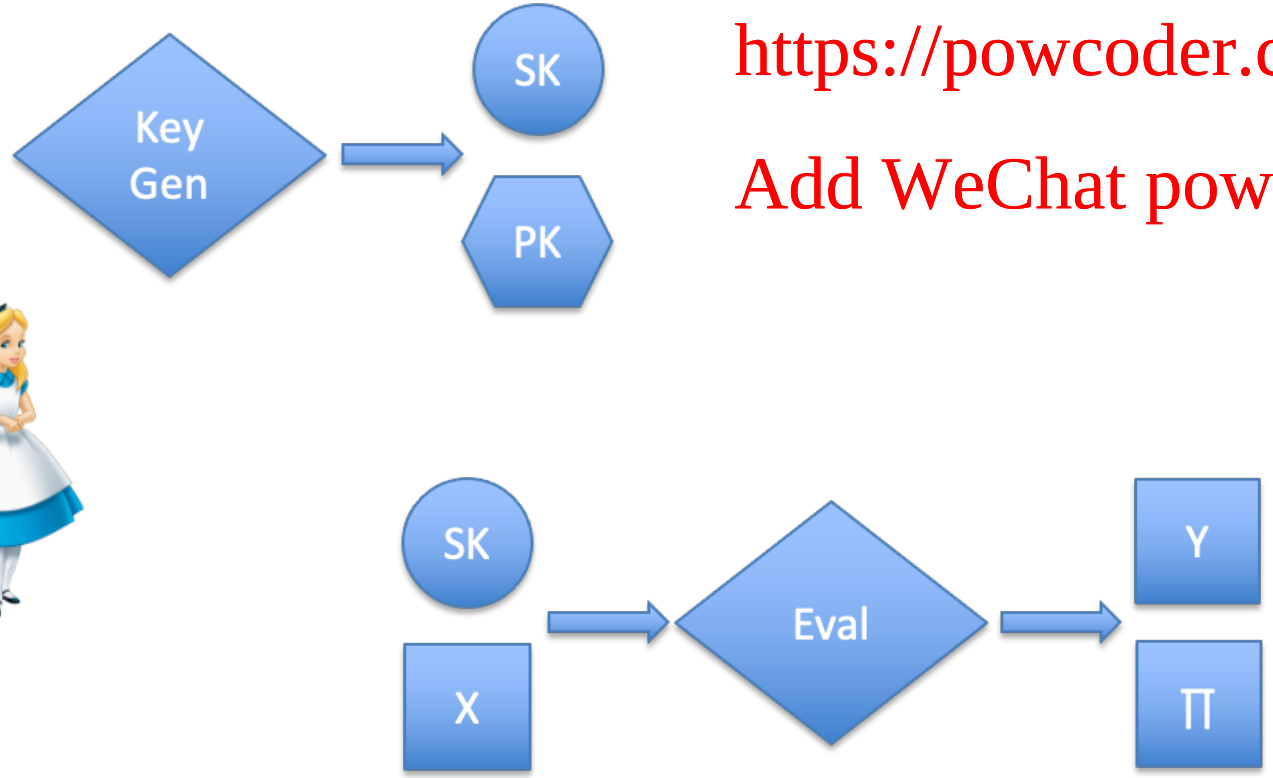
Verifiable Random Function (VRF)

- High-level idea: A PRF that can be publicly verified.



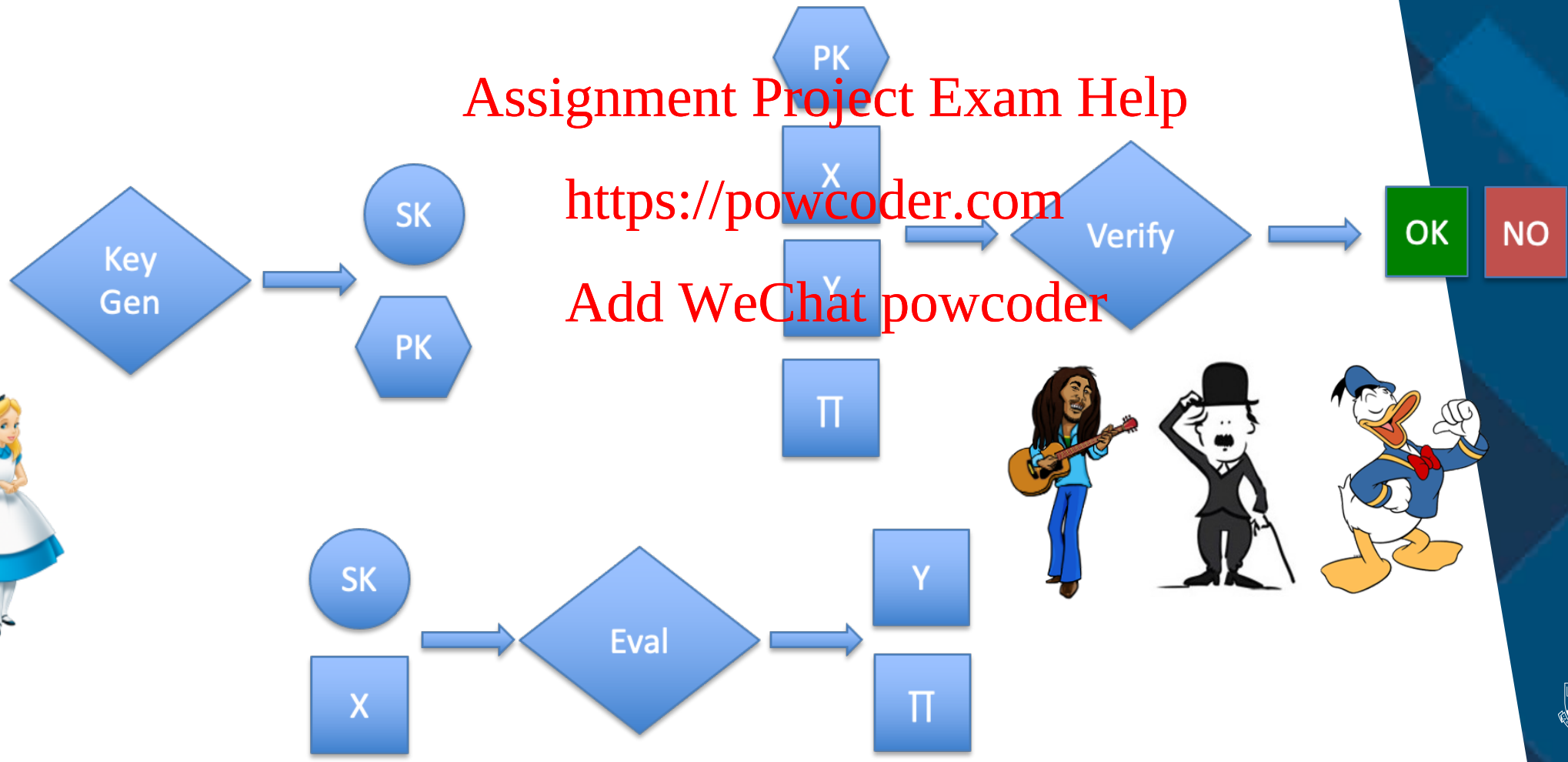
Verifiable Random Function (VRF)

- High-level idea: A PRF that can be publicly verified.



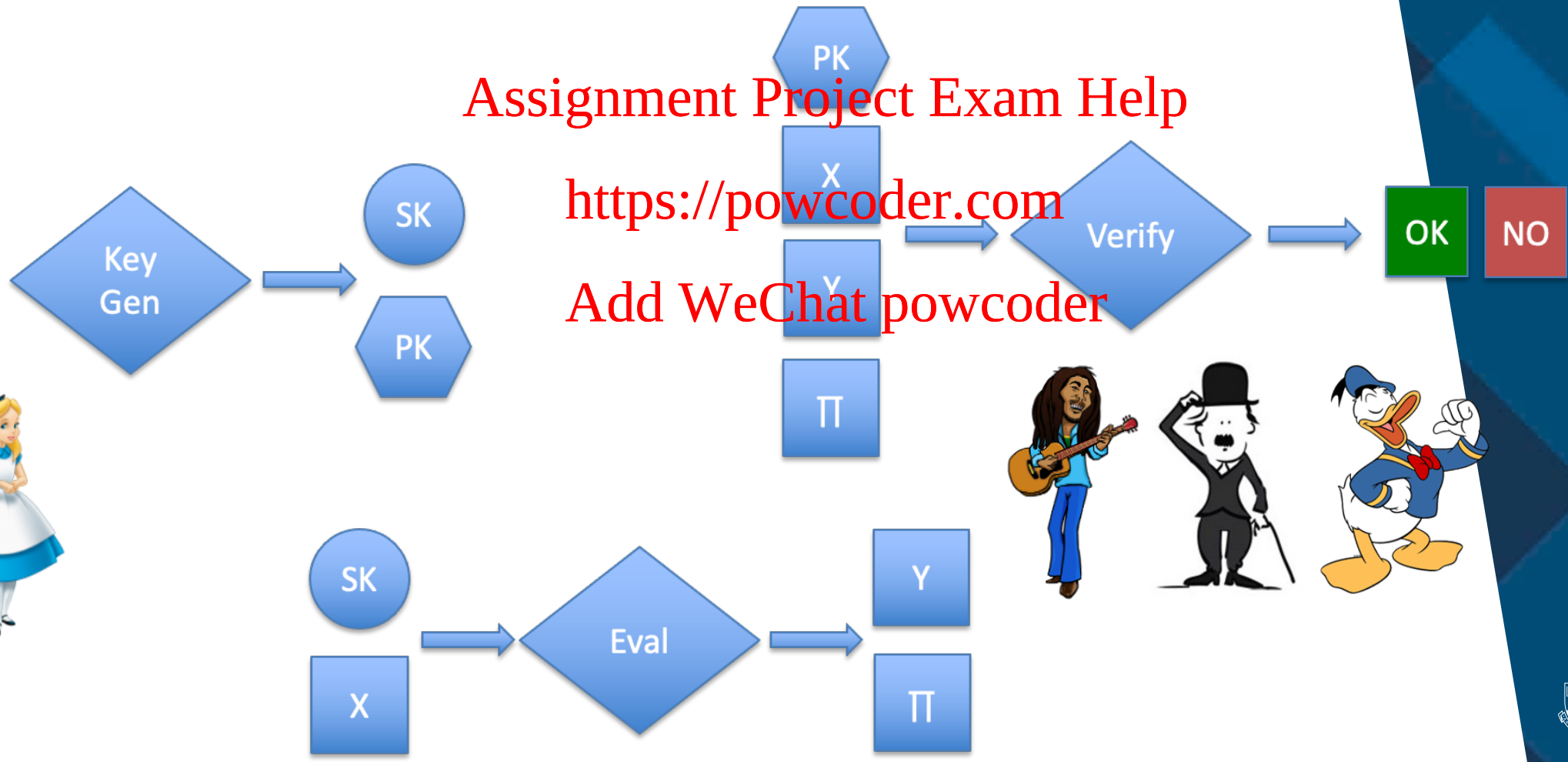
Verifiable Random Function (VRF)

- High-level idea: A PRF that can be publicly verified.



Verifiable Random Function (VRF)

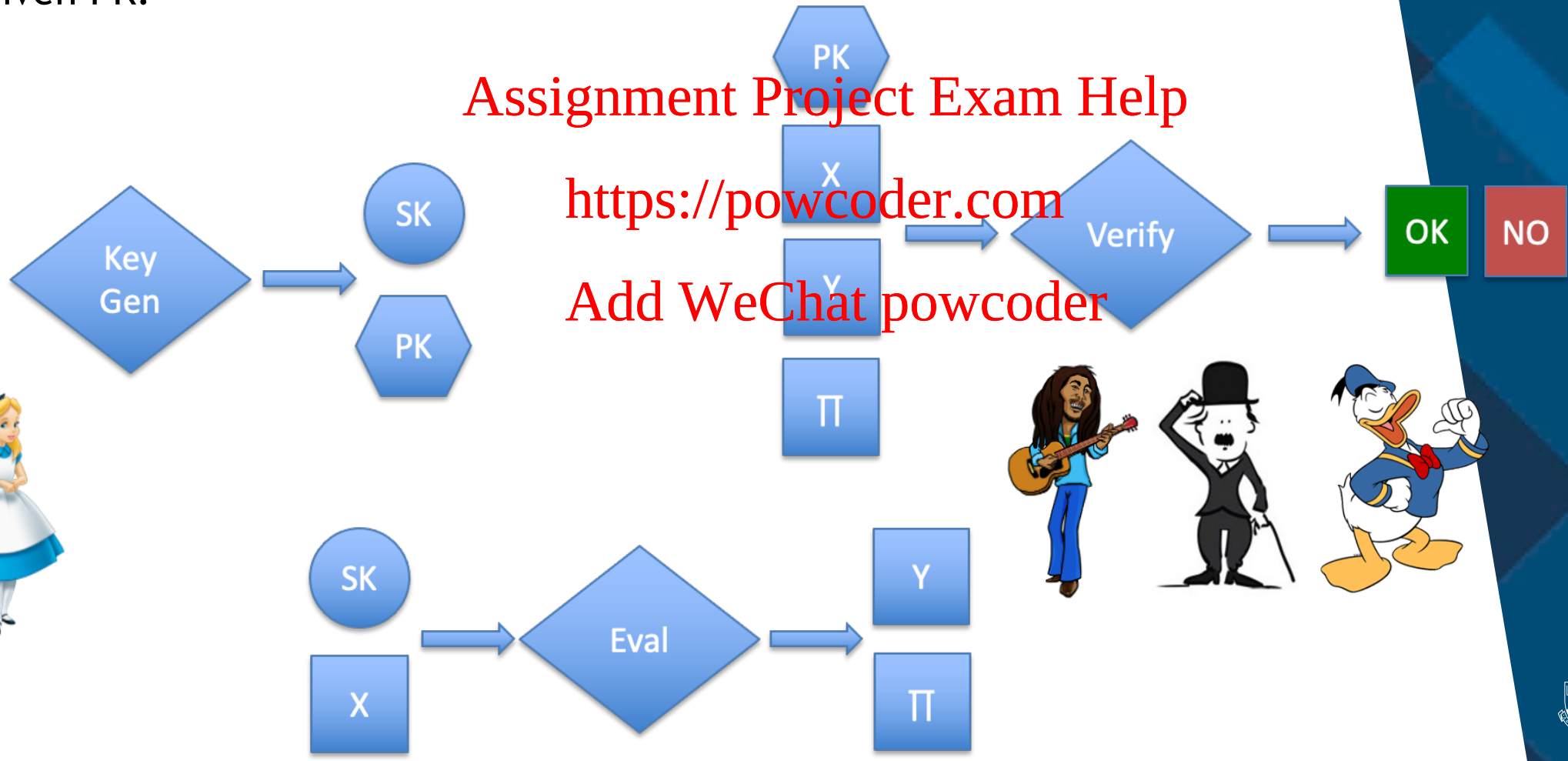
- Security properties of VRF: pseudorandomness, provability and uniqueness.



MONASH University

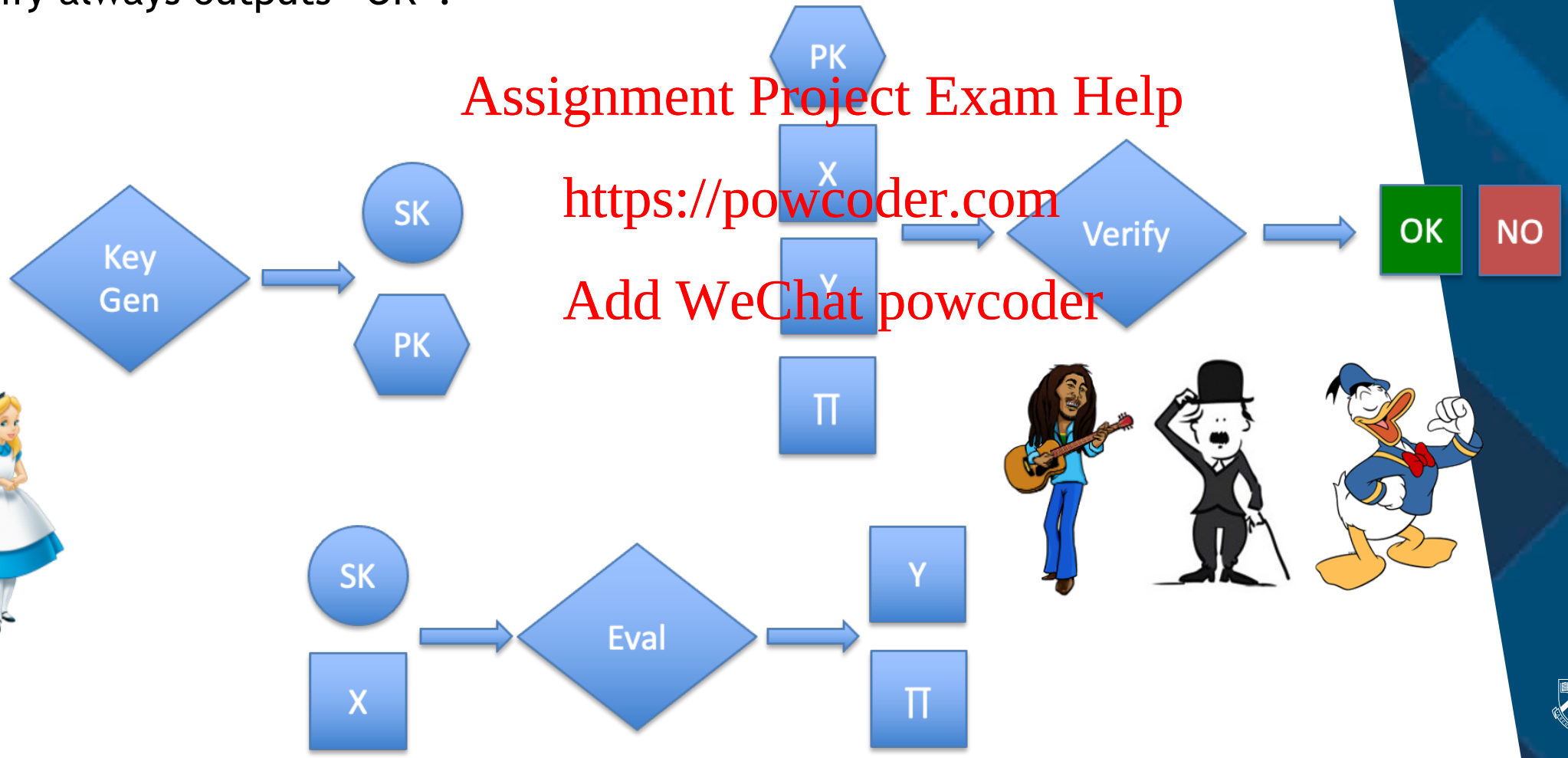
Verifiable Random Function (VRF)

- Pseudorandomness: the output Y is pseudorandom, even when the adversary is given PK.



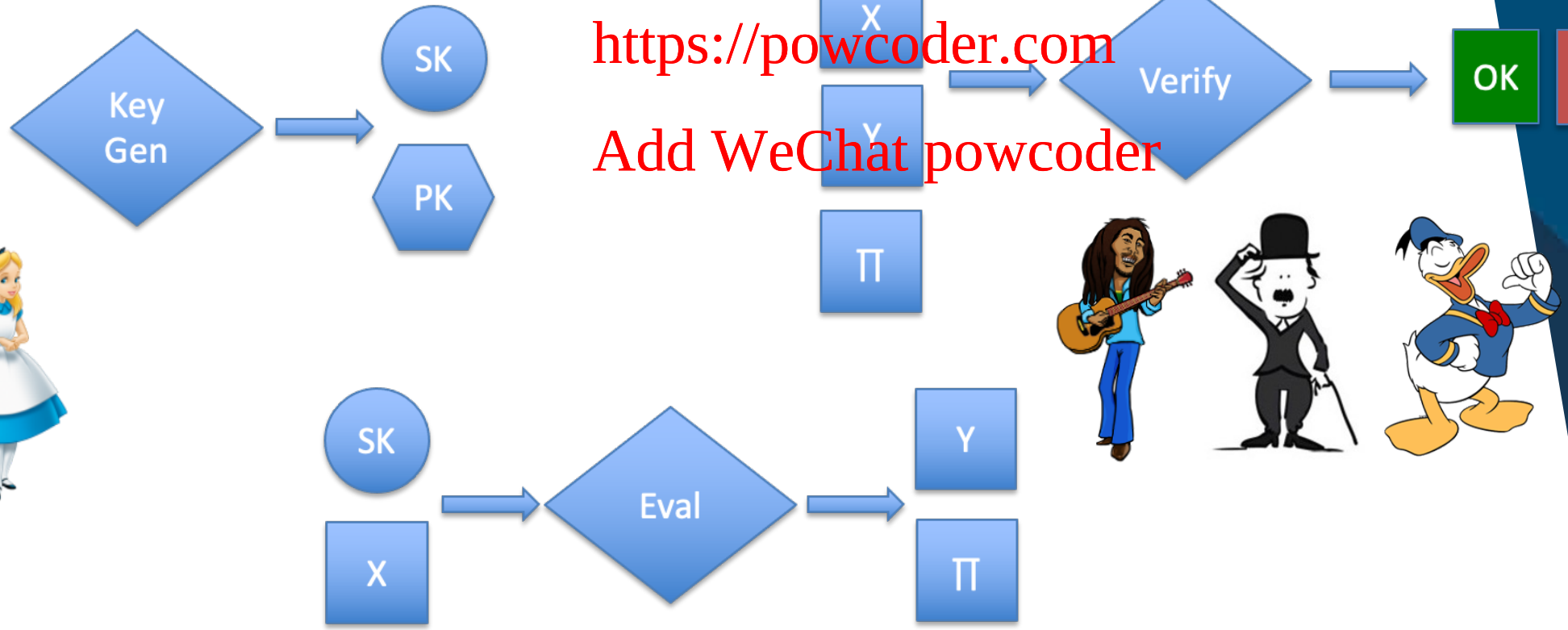
Verifiable Random Function (VRF)

- Provability: For honestly generated keys, input X, output Y and proof ∏, Verify always outputs “OK”.



Verifiable Random Function (VRF)

- Uniqueness: No tuple (PK, X, Y_0, Π_0, Y_1, Π_1) such that Verify(PK, X, Y_0,Π_0)=OK, Verify(PK, X, Y_1, Π_1)=OK and $Y_0 \neq Y_1$.



VRF in Ouroboros

- Special VRF that works correctly even under malicious key generation.
- Used in a local, private lottery for leader election.

$< T(stake_{User})$

VRF in Ouroboros

- Special VRF that works correctly even under malicious key generation.
- Used in a local, private lottery for leader election.

$< T(stake_{User})$

Assume for now that the parties have agreed on the random nonce rnd and stake distribution that are used above.

VRF in Ouroboros

- Special VRF that works correctly even under malicious key generation.
- Used in a local, private lottery for leader election.

$< T(stake_{User})$

Assume for now that the parties have agreed on the random nonce rnd and stake distribution that are used above.

Stake distribution and nonce that are used in the above computations remain unchanged for some amount of slots.

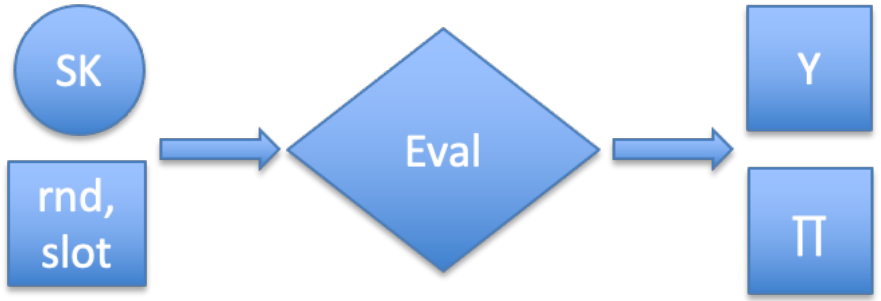
VRF in Ouroboros

- Special VRF that works correctly even under malicious key generation.
- Used in a local, private lottery for leader election.

- **Using a private lottery helps mitigating adaptive attacks.**
- When a user claims to be a slot leader, anyone can publicly verify if that claim is legitimate or not.

Slot Leader

- **Local, private lottery for leader election.**
- The slot leader is only revealed once he propagates a block to the network.
- Each user checks a local condition to determine if he has a legitimate claim for slot leadership.
- Some slots will have no leaders, some will have multiple.

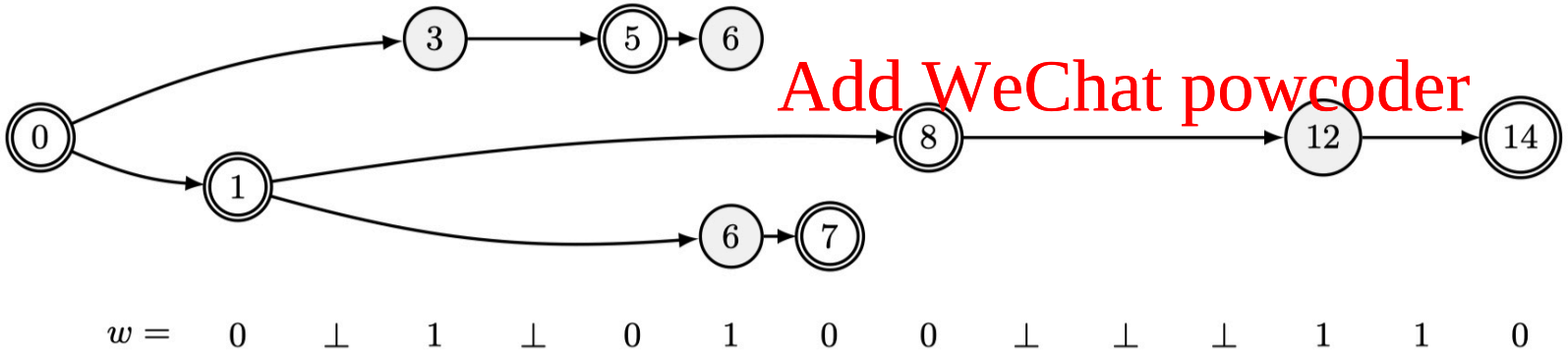


$< T(stake_{User})$

Characteristic Strings

Abstraction for analysing possible protocol executions.

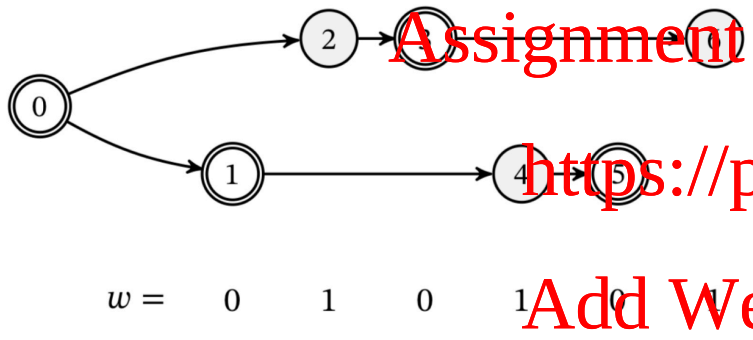
Characteristic string w:
- 0: slots with exactly one honest leader.
- 1: slots with a malicious leader or multiple leaders.
- ⊥: slots with no leader.



Such a string gives rise to a family of admissible graphs that describe all that can happen in an execution that follows the chain selection rule.

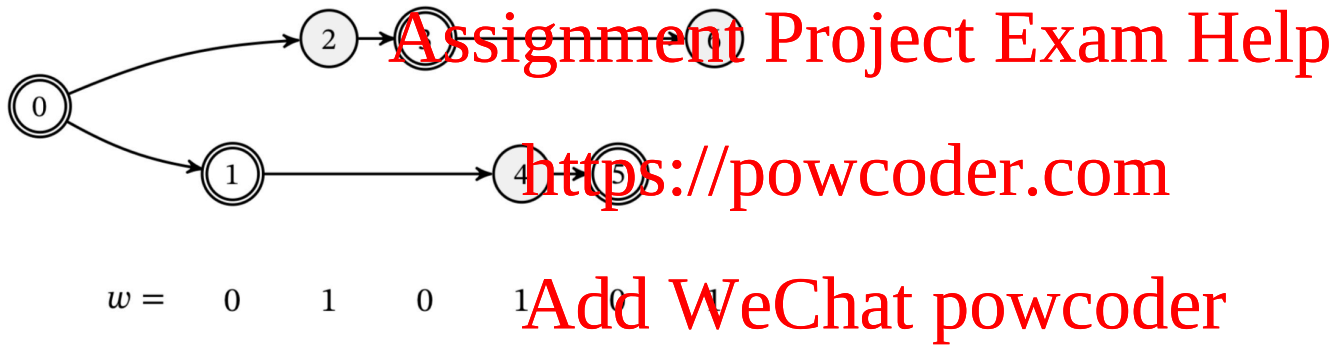
Forkable Strings

Forkable strings: characteristic strings that allow a fork with two tines of length equal to the height of the fork.



Forkable Strings

Forkable strings: characteristic strings that allow a fork with two tines of length equal to the height of the fork.



Forkable strings of big length are the problematic ones as they have a bad divergence point.

Common Prefix

The combinatorial divergence analysis that is presented in the Ouroboros papers proves that long forkable strings are a very unlikely structure to occur.

The details of this combinatorial analysis are beyond the scope of this unit. Interested students can find all details in the original papers.

Common Prefix

The combinatorial divergence analysis that is presented in the Ouroboros papers proves that long forkable strings are a very unlikely structure to occur.

The details of this combinatorial analysis are beyond the scope of this unit. Interested students can find all details in the original papers.

With no long diverging paths happening, the common prefix property is achieved.

Chain Quality

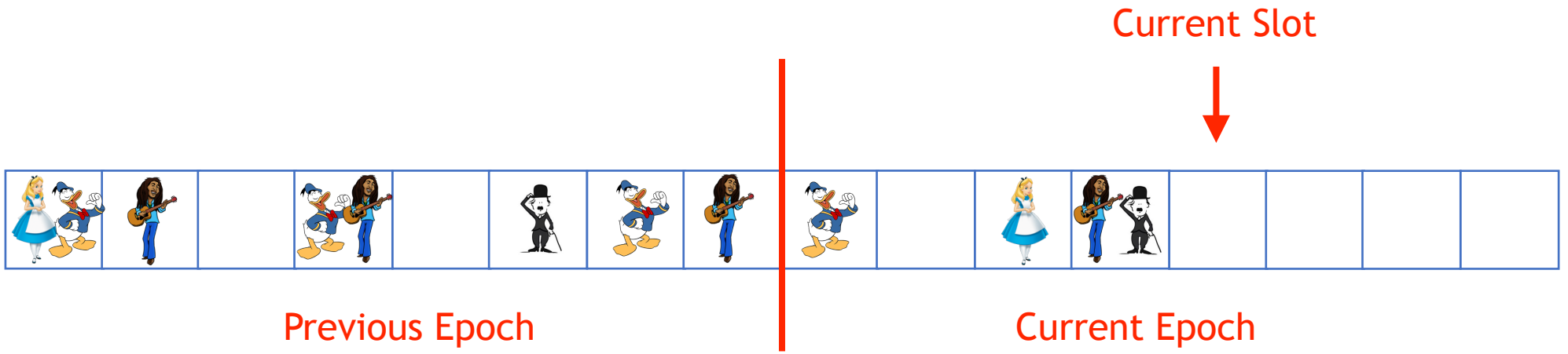
Any sufficiently long section along a (viable) tine must contain an honest node with overwhelming probability.

Chain Growth

The positions in which the characteristic string has a 0 support the growth of the chain, and then, by the chain quality property, growth is reflected in any viable tine.

Putting It All Together

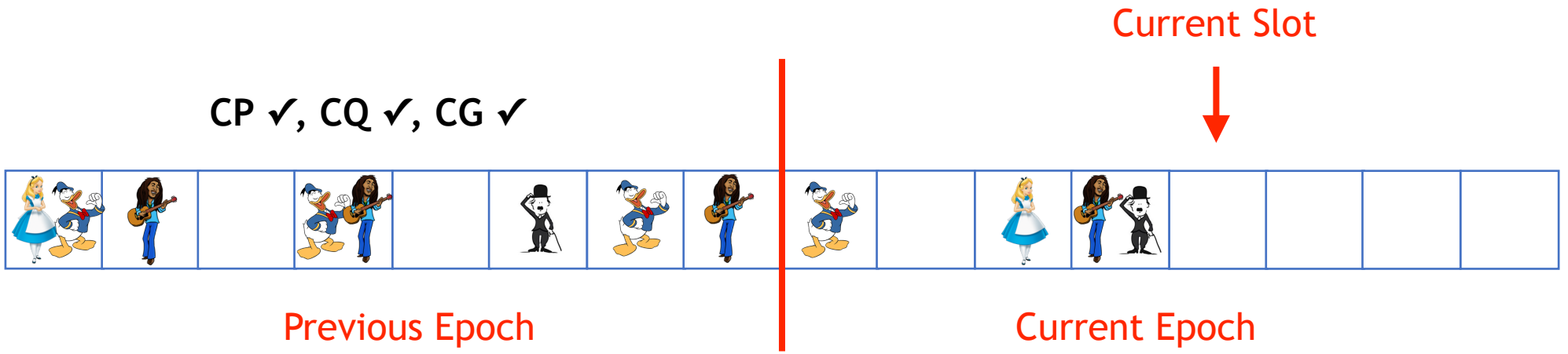
The protocols works in epochs. Each epoch contains a fixed number of slots.



Putting It All Together

The protocols works in epochs. Each epoch contains a fixed number of slots.

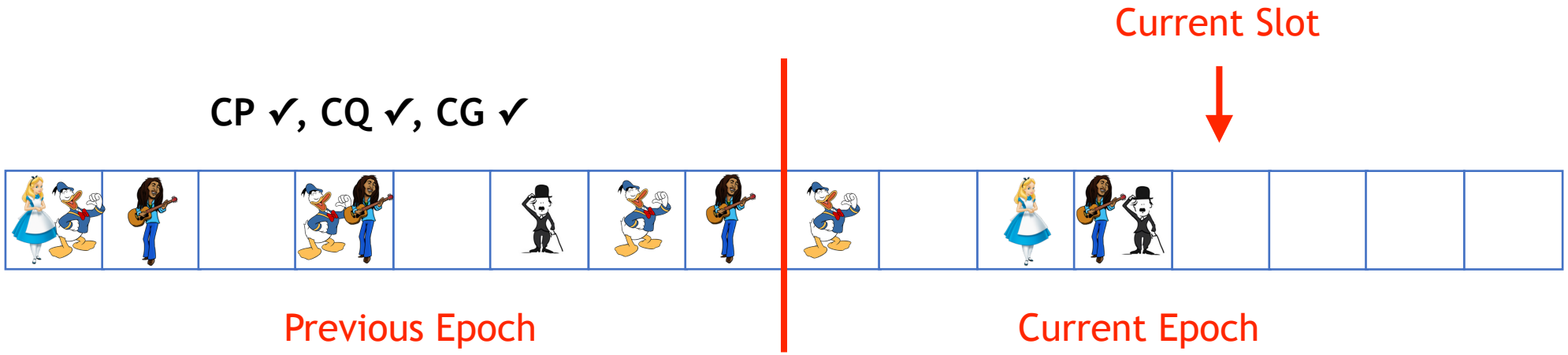
The Common Prefix, Chain Quality and Chain Growth properties hold for the previous epoch.



Putting It All Together

The protocols works in epochs. Each epoch contains a fixed number of slots.

The parties need to agree on the random nonce rnd and stake distribution that is used to select leaders in the current epoch.

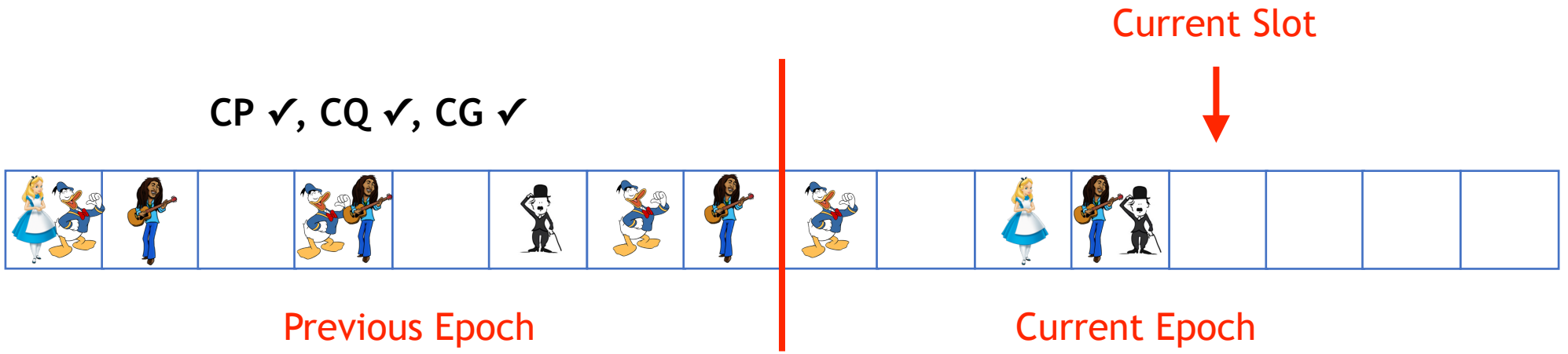


Putting It All Together

The protocols works in epochs. Each epoch contains a fixed number of slots.

The parties need to agree on the random nonce rnd and stake distribution that is used to select leaders in the current epoch.

For the first epoch this information comes directly from the genesis block. What about later epochs?

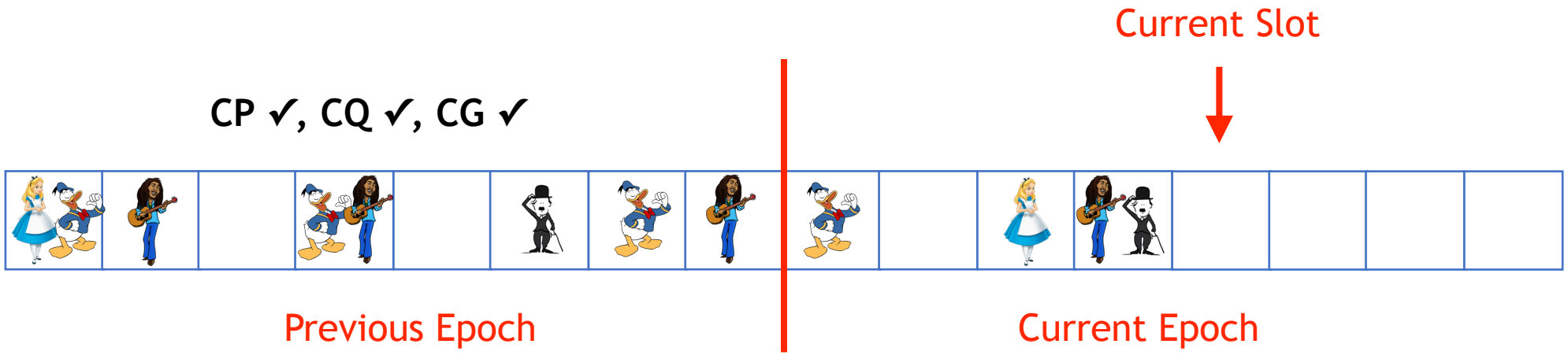


Putting It All Together

The protocols works in epochs. Each epoch contains a fixed number of slots.

The parties need to agree on the random nonce rnd and stake distribution that is used to select leaders in the current epoch.

Agreement on the stake distribution is done in the previous epoch and fixed for the current epoch.



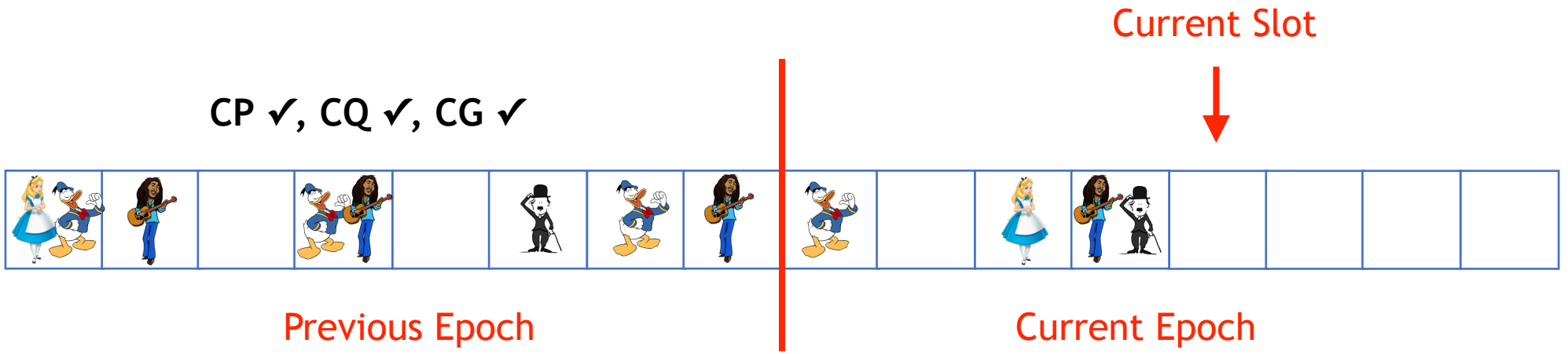
Putting It All Together

The protocols works in epochs. Each epoch contains a fixed number of slots.

The parties need to agree on the random nonce rnd and stake distribution that is used to select leaders in the current epoch.

Agreement on the stake distribution is done in the previous epoch and fixed for the current epoch.

Similarly, an agreement on the new random nonce is achieved on the previous epoch by hashing verifiable random values from the chain to obtain the new nonce. And this nonce is affected by honest blocks.



MONASH University

Additional Information

For further details about Ouroboros and Cardano, the following resources offer abundant information:

- IOHK Research Library
https://iohk.io/en/research/library/

- IOHK Youtube Channel
https://www.youtube.com/channel/UCBJ0p9aCW-W82TwNM-z3V2w

Algorand is another very interesting solution combining balanced-based PoS (also using VRFs) and Byzantine Agreement:
https://www.algorand.com/technology/white-papers

Next Week

- Privacy