Blockchain Principles and Applications

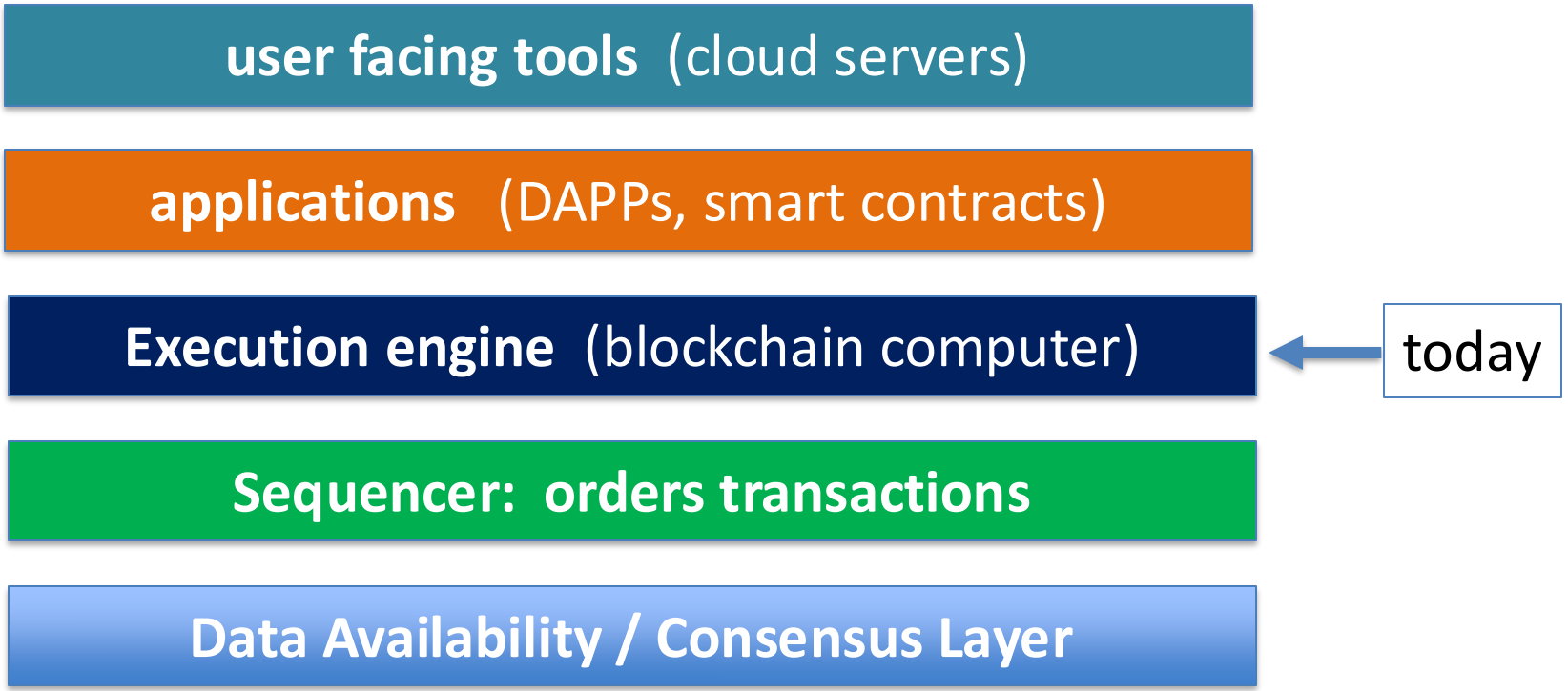
Amir Mahdi Sadeghzadeh, PhD

Data and Network Security Lab (DNSL)
Trustworthy and Secure AI Lab (TSAIL)

Most slides have been adapted from ECE/COS 470, Pramod Viswanath, Princeton 2024 and CS251, Dan Boneh, Stanford 2023.

Recap

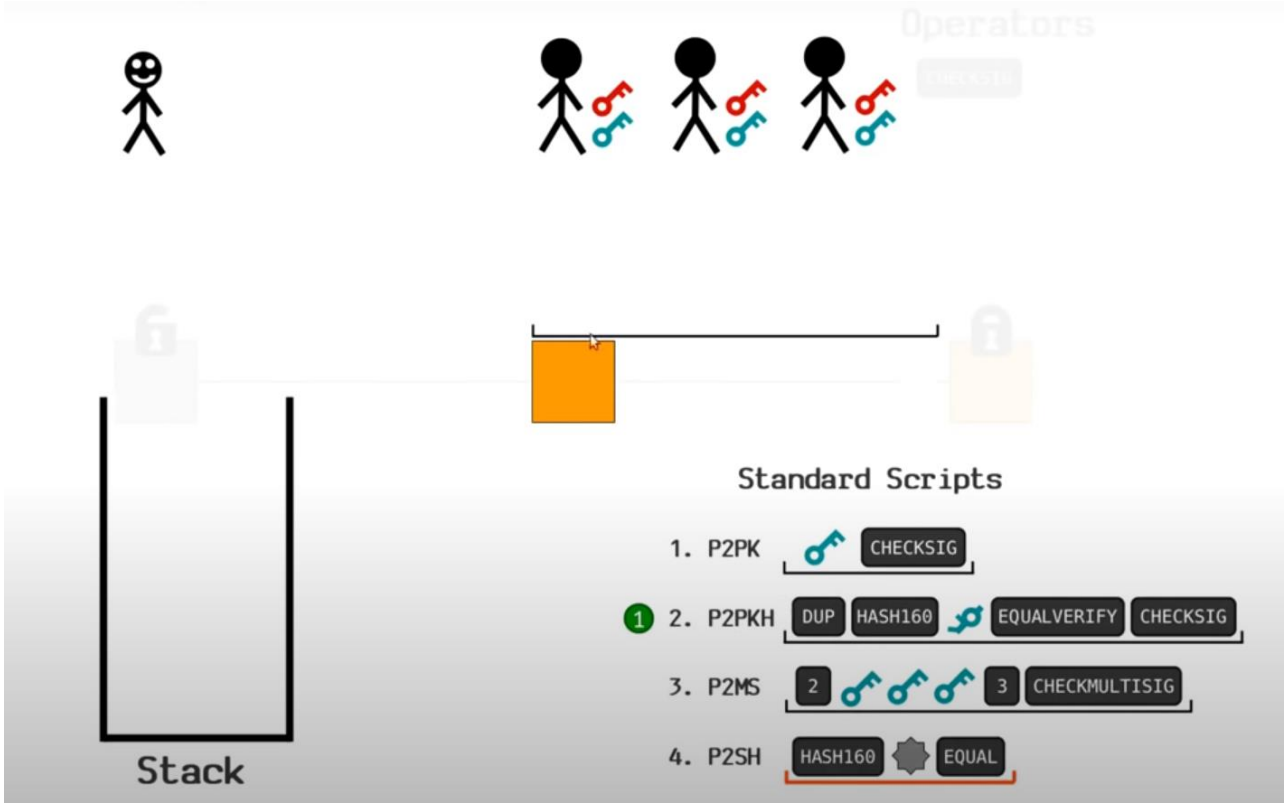
This lecture: Bitcoin mechanics



What is Script?

- Script is a mini programming language used as a locking mechanism for outputs in bitcoin transactions.
  - A locking script (ScriptPubKey) is placed on every transaction output.
  - An unlocking script (ScriptSig or Witness) must be provided to unlock an output (i.e. when used as an input to a transaction).
- If a full script (unlocking + locking) is valid, the output is "unlocked" and can be spent.

Pay to Script Hash (P2SH)



Example: a common script

<sig> <pk>	**DUP HASH256** <pkhash> **EQVERIFY CHECKSIG**

stack:	
empty	init
<sig> <pk>	push values
<sig> <pk> <pk>	**DUP**
<sig> <pk> <hash>	**HASH256**
<sig> <pk> <hash> <pkhash>	push value
<sig> <pk>	**EQVERIFY**
1	**CHECKSIG**

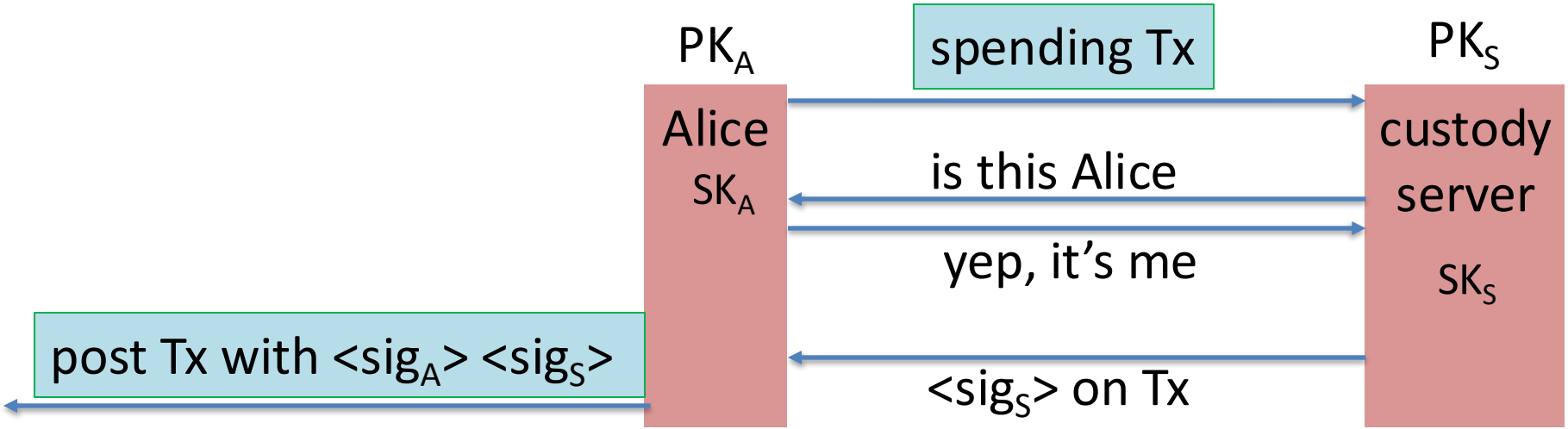
verify(pk, Tx, sig)

⇒ successful termination

Example Bitcoin scripts

Protecting assets with a co-signatory

Alice stores her funds in UTXOs for $addr$ = **2-of-2(PK_A, PK_S)**



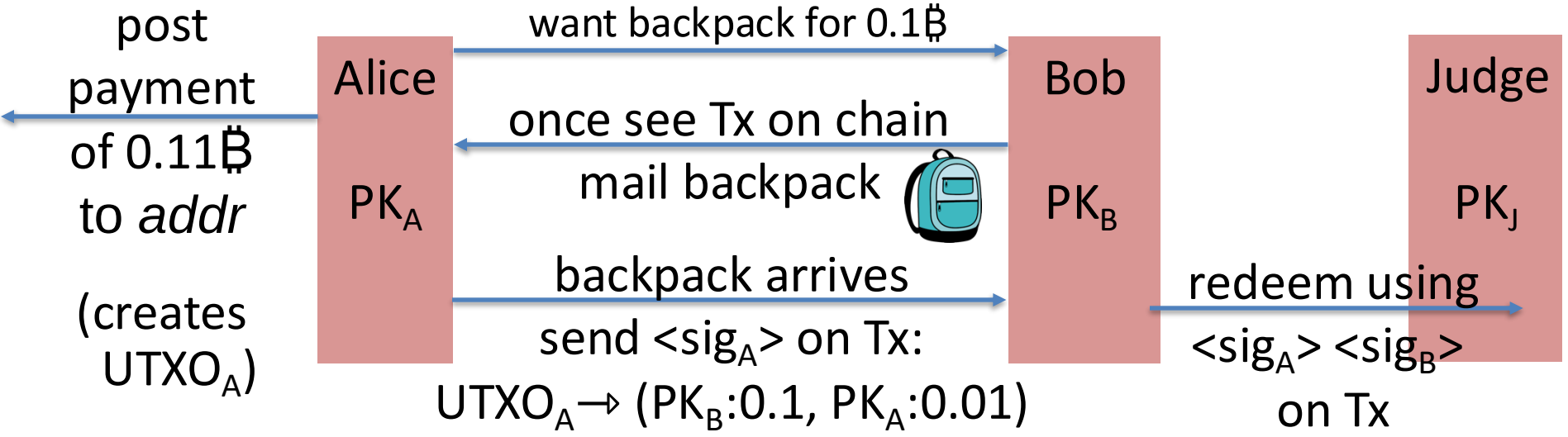
⇒ theft of Alice's SK_A does not compromise BTC

Escrow service

Alice wants to buy a backpack for 0.1฿ from merchant Bob

Goal: Alice only pays after backpack arrives, but can't not pay

$$addr = \textbf{2-of-3}(PK_A, PK_B, PK_J)$$



Escrow service: a dispute

(1) Backpack never arrives: (Bob at fault)

Alice gets her funds back with help of Judge and a Tx:

Tx: $\mathbf{(UTXO_A \rightarrow PK_A \ , \ sig_A, \ sig_{Judge} \ )}$ [2-out-of-3]

---

(2) Alice never sends sig_A: (Alice at fault)

Bob gets paid with help of Judge and a Tx:

Tx: $\mathbf{(UTXO_A \rightarrow PK_B \ , \ sig_B, \ sig_{Judge} \ )}$ [2-out-of-3]

---

(3) Both are at fault: Judge publishes $<sig_{Judge}>$ on Tx:

Tx: $\mathbf{(UTXO_A \rightarrow PK_A: 0.05, \ PK_B: 0.05, \ PK_J: 0.01 \ )}$

Now either Alice or Bob can execute this Tx.

Bitcoin Safety

Bitcoin Security

- **Safety**: A transaction/block confirmed by one user is soon confirmed by all other users and remains confirmed forever after.

- **Liveness**: all (honest) transactions get included into blocks, and further that the blocks feature in the longest chain.

Spam protection

Truly permissionless: anyone can join and do anything

Network data: transactions and blocks
Both data types have inbuilt cryptographic resistance to spam

- Transaction: digital signature
- Blocks: PoW & syntax of the header

Protocol level attacks

√ Create valid blocks

x Mine on the tip of the longest chain

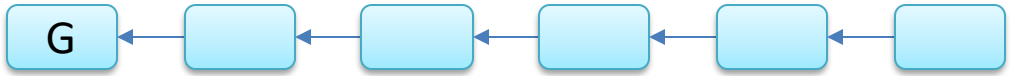
x Publish the blocks once mined

We looked at one strategy called private attack

Longest Chain Protocol

Where should the mined block hash-point to?

Latest block?



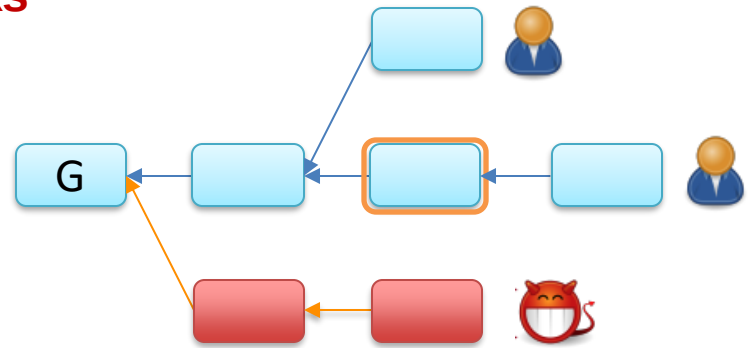
Longest Chain Protocol

Where should the mined block hash-point to?

However, blockchain may have **forks**

because of network delays

because of adversarial action



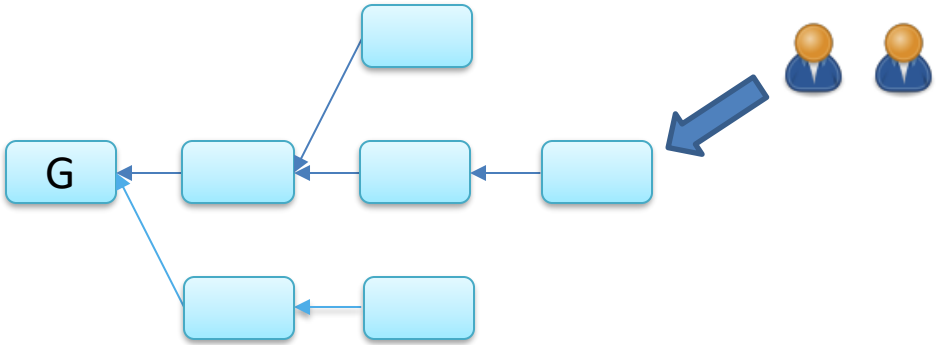
Longest Chain Protocol

Where should the mined block hash-point to?

Blockchain may have **forks**

because of network delays

because of adversarial action



Longest chain protocol

attach the block to the leaf of the longest chain in the block tree

Double Spend Attack

Adversary can point its block to an older part of the chain

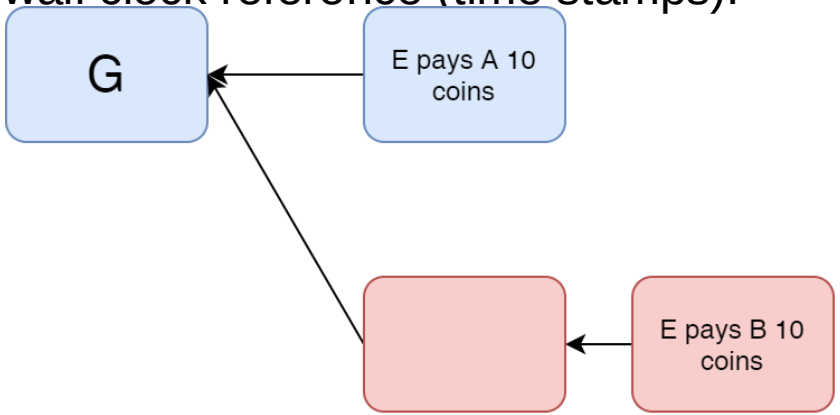
Duplicate transaction inserted

Plausible Deniability

network latency

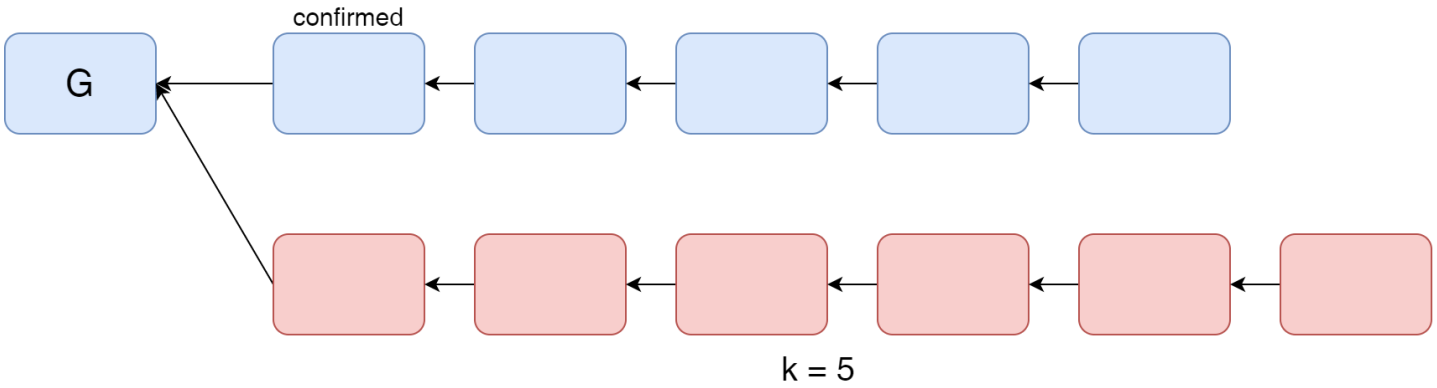
an offline user will not know which block came earlier

blocks have no wall clock reference (time stamps).



k Deep Confirmation Rule

- A block is **confirmed** if it is **buried k-deep in the longest chain**
- An attacker would need more than k blocks to double spend



Stochastic Process

A **stochastic process** is a mathematical model that describes a collection of random variables indexed by time (or another parameter), representing how a system evolves under uncertainty. It is widely used in probability theory, statistics, and applications like finance, physics, and machine learning.

Definition

A stochastic process is a family of random variables $\{X(t) : t \in T\}$, where:

- T is the index set (often representing time, either discrete or continuous).
- $X(t)$ is a random variable representing the system state at time t.
- The randomness means that each observation of the process may yield different outcomes.

Poisson process

A Poisson process $N(t)$ with rate (or intensity) λ satisfies the following properties:

1. **Independent increments**: The number of events occurring in disjoint time intervals are independent.
2. **Stationary increments**: The number of events occurring in any interval of length t follows a Poisson distribution with mean λt:

$$P(N(t) = k) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}, \quad k = 0, 1, 2, \dots$$

3. **No simultaneous events**: The probability of more than one event occurring in an infinitesimally small interval $[t, t + \Delta t]$ is negligible:

$$P(N(t + \Delta t) - N(t) = 1) \approx \lambda \Delta t, \quad P(N(t + \Delta t) - N(t) \geq 2) \approx 0.$$

Poisson process

Interpretation of λ

- λ represents the average number of events occurring per unit time.
- The **interarrival times** between consecutive events are exponentially distributed with mean $1/\lambda$, meaning the waiting time between events follows:

$$P(T > t) = e^{-\lambda t}, \quad t \geq 0.$$

Applications

- **Network traffic**: Packet arrivals in a network often follow a Poisson process.
- **Reliability analysis**: Failures of a system might be modeled as a Poisson process.
- **Queueing theory**: Customers arriving at a service counter can be modeled by a Poisson process.
- **Finance**: Modeling the arrival of trades or price changes in high-frequency trading.

Exponential distribution

Probability density function

The probability density function (pdf) of an exponential distribution is

$$f(x;\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases}$$

Here $\lambda > 0$ is the parameter of the distribution, often called the *rate parameter*. The distribution is supported on the interval $[0, \infty)$. If a

The **cumulative distribution function (CDF)** of an **exponential distribution** with rate parameter $\lambda > 0$ is given by:

$$F(t) = P(T \leq t) = \begin{cases} 1 - e^{-\lambda t}, & t \geq 0 \\ 0, & t < 0 \end{cases}$$

Exponential distribution

Probability density function

The probability density function (pdf) of an exponential distribution is

$$f(x;\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases}$$

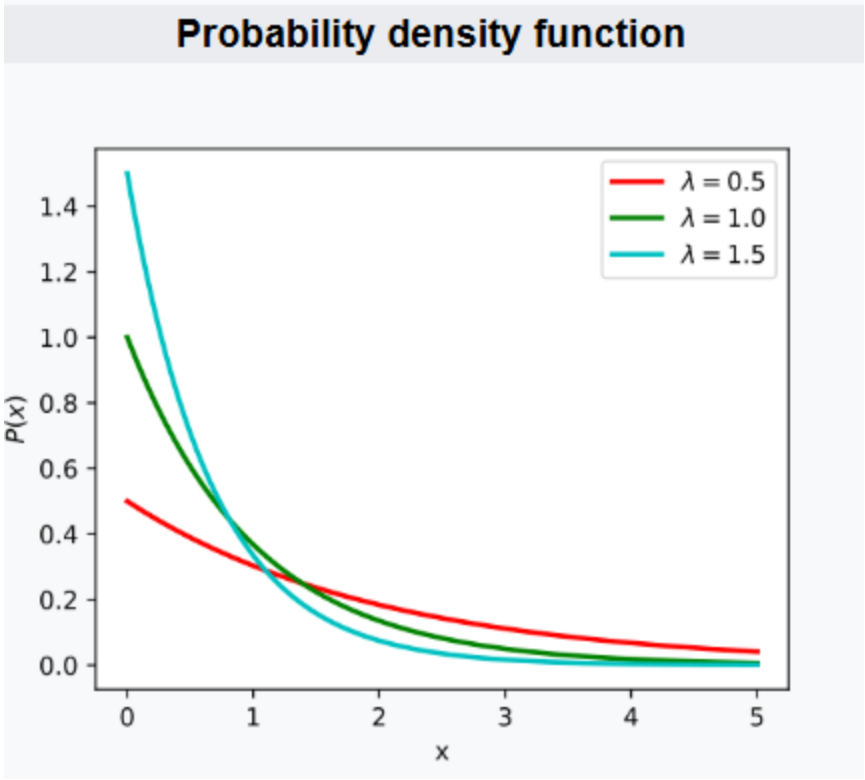
Here $\lambda > 0$ is the parameter of the distribution, often called the *rate parameter*. The distribution is supported on the interval $[0, \infty)$.

Recall:

The mean or expected value of an exponentially distributed random variable X with rate parameter λ is given by

$$\mathrm{E}[X] = \frac{1}{\lambda}.$$

Exponential distribution



Mining as a Poisson process

The times at which a new block is mined is modeled as a Poisson process with rate λ. Here the average inter-block time, $\frac{1}{\lambda}$, is set based on the target difficulty in the PoW mining operation; for Bitcoin $\frac{1}{\lambda} = 10$ minutes. A Poisson process is one in which new events (or arrivals) occur at random intervals following the exponential distribution. Moreover, the intervals between any two events are independent of, and statistically identical to, each other. Recall that an exponential random variable X with parameter λ has distribution

$$\mathbb{P}(X \geq t) = \exp(-\lambda t) \ \forall t \geq 0.$$

Memorylessness property of exponential random variable

An exponentially distributed random variable T obeys the relation

$$\Pr\left(T > s + t \mid T > s\right) = \Pr(T > t), \qquad \forall s, t \geq 0.$$

This can be seen by considering the complementary cumulative distribution function:

$$\begin{aligned}
\Pr\left(T > s + t \mid T > s\right) &= \frac{\Pr\left(T > s + t \cap T > s\right)}{\Pr\left(T > s\right)} \\
&= \frac{\Pr\left(T > s + t\right)}{\Pr\left(T > s\right)} \\
&= \frac{e^{-\lambda(s+t)}}{e^{-\lambda s}} \\
&= e^{-\lambda t} \\
&= \Pr(T > t).
\end{aligned}$$

Mining as a Poisson process

Also recall that a Poisson random variable Y with parameter λ has distribution

$$\mathbb{P}(Y = k) = \exp(-\lambda)\frac{\lambda^k}{k!} \ \forall k \geq 0.$$

In a Poisson process, the number of events in an interval of length T is a Poisson random variable with parameter λT. Moreover, the number of events in disjoint intervals of time are independent. If we consider small intervals ($\lambda T \ll 1$), there is one event in the interval with probability λT and none otherwise. Thus, a Poisson process can be emulated by counting the occurrence of heads in a sequence of (independent) coin tosses, with the probability of heads being very small. We now see why the mining process has such a property.

Mining as a Poisson process

In modeling the mining process as a Poisson process, we focus only at the times at which new, valid blocks are created. The Poisson process model holds irrespective of the number of miners, their individual computation power, whether different miners are working on the same block or different blocks, and when different users receive newly mined blocks. The parameter λ of the mining process, called the **mining rate**, is equal to the average number of blocks mined per unit time. In Bitcoin, λ is $1/(600\text{s})$, i.e., one block every 600 seconds (ten minutes). In Ethereum (which also has the same Nakamoto consensus protocol), the rate is much faster: λ is $1/(13\text{s})$.

Mining as a Poisson process

We assume that there is a single adversary and many different honest parties participating in the protocol. The adversary's computing power is a fraction β of the total computing power of all users in the system. Among honest users, the computing power is divided roughly equally with each user controlling a very small fraction. This implies that typically, consecutive honest blocks are mined by different users. Such a model gives the adversary more power than a setting with a smaller number of honest users with considerable mining power. This will be made clear when we discuss the effect of network delay. Let the fraction of "hash power" of the adversary be β, and assume that $\beta < 1/2$. This means that the adversary mines blocks as a Poisson process of rate $\beta\lambda$, while honest users mine blocks at a rate of $(1 - \beta)\lambda$. Further, these processes are independent of each other.

Nakamoto consensus protocol model

they have heard until that time. Let $\mathcal{C}_i^h$ denote the chain held by party h at time $i \in \mathbb{R}^+$.

If an honest party hears of multiple chains with the same (maximum) length, we assume that they choose one of them arbitrarily as $\mathcal{C}_i^h$. This choice is made by the adversary. Note that this implies an honest user may swap its chain when it hears of an equally long chain, not just a strictly longer one. It also implies that if two honest parties both hear of two chains of equal (maximum) length, then the two parties may adopt different chains. This tie breaking power is another example of giving the adversary extra powers than what may exist in reality.

Nakamoto consensus protocol model

The *prefix* of a chain is a sub-chain consisting of the first few blocks. More formally, we say that chain $\mathcal{C}_1$ is a prefix of chain $\mathcal{C}_2$ if all blocks in $\mathcal{C}_1$ are also present in $\mathcal{C}_2$. (By default, we assume that a chain is a sequence of blocks from the genesis down to any other block.) We denote this by $\mathcal{C}_1 \preceq \mathcal{C}_2$. We define the notation of the prefix of a chain as follow.

- For a chain $\mathcal{C}$, let $\mathcal{C}^{\lfloor k}$ be the prefix chain obtained by dropping the last k blocks. In case $\mathcal{C}$ has less than or equal to k blocks, let $\mathcal{C}^{\lfloor k}$ be the genesis block.

Nakamoto consensus protocol model

Recall the k-deep confirmation rule in Bitcoin: a node treats all but the last k blocks in its longest chain as *confirmed*. In our notation, the blocks in $\mathcal{C}_i^{h\lfloor k}$ are confirmed by user h at time i. Once we confirm a block, we also confirm all the transactions in it. Note that the confirmation rule is *locally* applied by each user; therefore, a transaction confirmed by one user needn't be confirmed by another. However, it is desirable that a transaction confirmed by one user is soon confirmed by all other users, and remains confirmed forever after. In blockchains, this desirable property is called a *safety property*.

Formal definitions of safety

Definition 1 (Common Prefix Property). *For a blockchain protocol, the k-common prefix property holds during an execution of the protocol if any block that is committed by one honest user appears in every honest user's chain thereafter. Mathematically, for all pairs of times $i_1 \leq i_2$, for all pairs of honest users h_1, h_2,*

$$\mathcal{C}_1^{\lfloor k} \preceq \mathcal{C}_2$$

where $\mathcal{C}_1 \equiv \mathcal{C}_{i_1}^{h_1}$, $\mathcal{C}_2 \equiv \mathcal{C}_{i_2}^{h_2}$.

Formal definitions of safety

Definition 2 (Individual block safety). *In an execution, a block B present in some honest user's chain is safe if, after it has been committed by any honest user, it remains a part of all honest user's chains. Mathematically, if block B is committed by some user h at time t, then for all $t' \geq t$, for all honest users h', $b \in \mathcal{C}_{t'}^{h'}$*

Mining as a Poisson process

Time to a successful mining event is an **exponential** random variable

$$T \sim exp(\lambda) \;\; if \;\; \Pr(T \geq t) = e^{-\lambda t}$$

Memoryless:

$$\Pr(T \geq t + t_0 | T \geq t_0) = \frac{\Pr(T \geq t + t_0)}{\Pr(T \geq t_0)} = \frac{e^{-\lambda(t+t_0)}}{e^{-\lambda t_0}} = e^{-\lambda t} = \Pr(T \geq t)$$

Number of mined blocks in time T is a **Poisson** random variable

$$X \sim Poi(\lambda T) \;\; if \;\; \Pr(X = k) = \frac{(\lambda T)^k e^{-\lambda T}}{k!}$$

The mining process is a Poisson process with rate λ, proportional to hash power

Mining as a Poisson Process

Mathematical fact: The sum of multiple independent Poisson processes is still a Poisson process

Consequence: the honest/adversarial mining processes are independent Poisson processes with constant mining rate

Honest mining: Poisson process with rate $(1-\beta)\lambda$

Adversarial mining: Poisson process with rate $\beta\lambda$

Sum of Two Independent Poisson Processes

Let:

- $N_1(t)$ be a Poisson process with rate λ_1.
- $N_2(t)$ be a Poisson process with rate λ_2.

If $N_1(t)$ and $N_2(t)$ are independent, then their sum:

$$N(t) = N_1(t) + N_2(t)$$

is also a **Poisson process** with rate:

$$\lambda = \lambda_1 + \lambda_2.$$

Private attack

Consider a simple model of the blockchain system with all users split into two groups: many honest users and a singe adversarial user. The adversarial user is trying to re-write the last k blocks in the ledger by performing a private attack. Assume that the total mining rate is fixed at λ. We suppose that the adversarial fraction of hash power is β (for "bad"); the honest fraction of hash power is $1-\beta$.

Suppose the adversary wishes to violate the safety of a block B. To do so, the adversary must ensure that block B is first confirmed by some (or all) honest users, and then, at some time in the future, must dislodge B from the longest chain, i.e., it must create a fork from a block preceding B, and must eventually produce a chain of length equal to or longer than the longest chain containing B, after B has been confirmed.

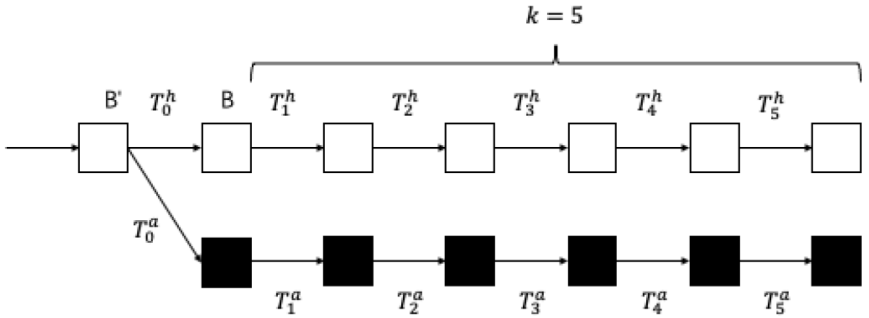


Figure 1: Private attack on block B with $k = 5$.

Private attack

One possible attack is the private attack, introduced in Lecture 3, which we investigate here in more detail. Let B' be the parent of block B. One option is for the adversary to mine a conflicting block on B' immediately after block B' is mined. It keeps mining in private, creating an ever-increasing chain. The honest users, unaware of the private chain, continue to mine following the longest chain rule, below B. In the private attack, the adversary does not contribute to the chain the honest nodes are mining on. Note that the honest and adversarial chains are independent of each other and distributed as Poisson processes with rate $(1-\beta)\lambda$ and $\beta\lambda$, respectively.

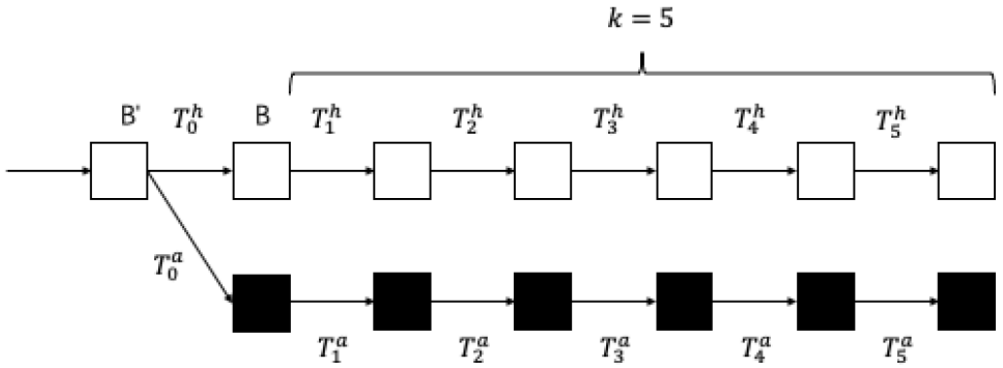


Figure 1: Private attack on block B with $k = 5$.

Private attack

When should the adversary reveal its chain to the honest users? Suppose it reveals its private chain while it is shorter than the longest honest chain. The honest users will simply ignore the adversary's chain, and it will not produce any effect. Thus, the adversary must reveal its chain only when it is at least as long as the honest chain. What happens if the adversary reveals its chain too early, i.e., before block B gets k-deep? It would still end up displacing B, but then it's actions do not lead to a safety violation! Thus, the adversary must also wait until the honest chain is long enough (see Figure 1).

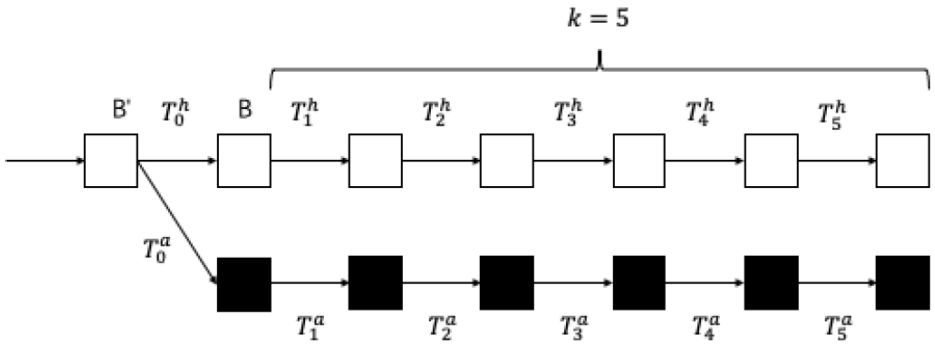
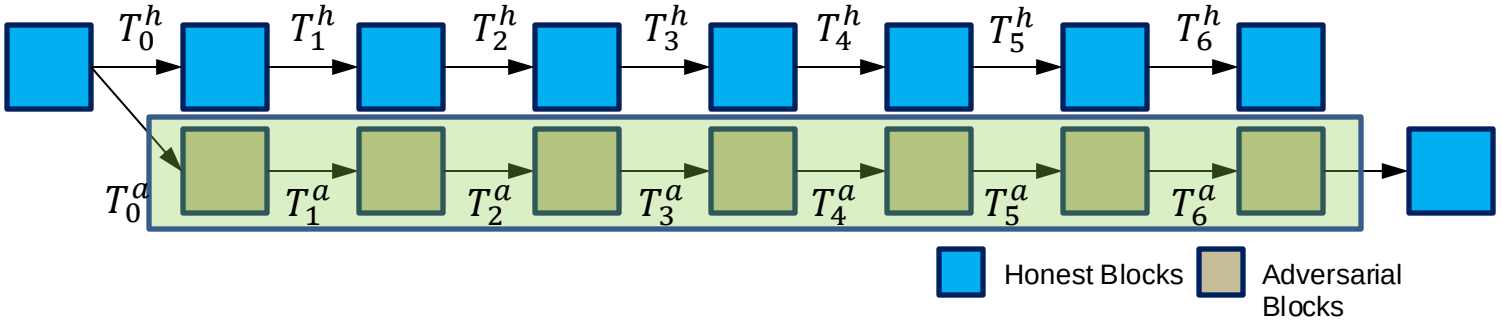


Figure 1: Private attack on block B with $k = 5$.

Private attack

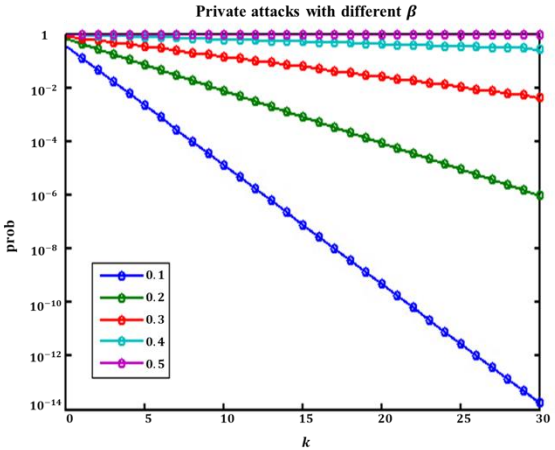
Private Attack



Attack Success

$k \to \infty \Rightarrow \sum_{i=0}^{\infty} T_i^h > \sum_{i=0}^{\infty} T_i^a \to \beta\lambda > (1-\beta)\lambda \to \beta > \frac{1}{2}$

$k < \infty \Rightarrow p_a = e^{-c(k+1)}$



Resources

- ECE/COS 470, Pramod Viswanath, Princeton 2024 (notes)
- CS251, Dan Boneh, Stanford 2023
- Bitcoin Lesson – Transactions by learnmebitcoin