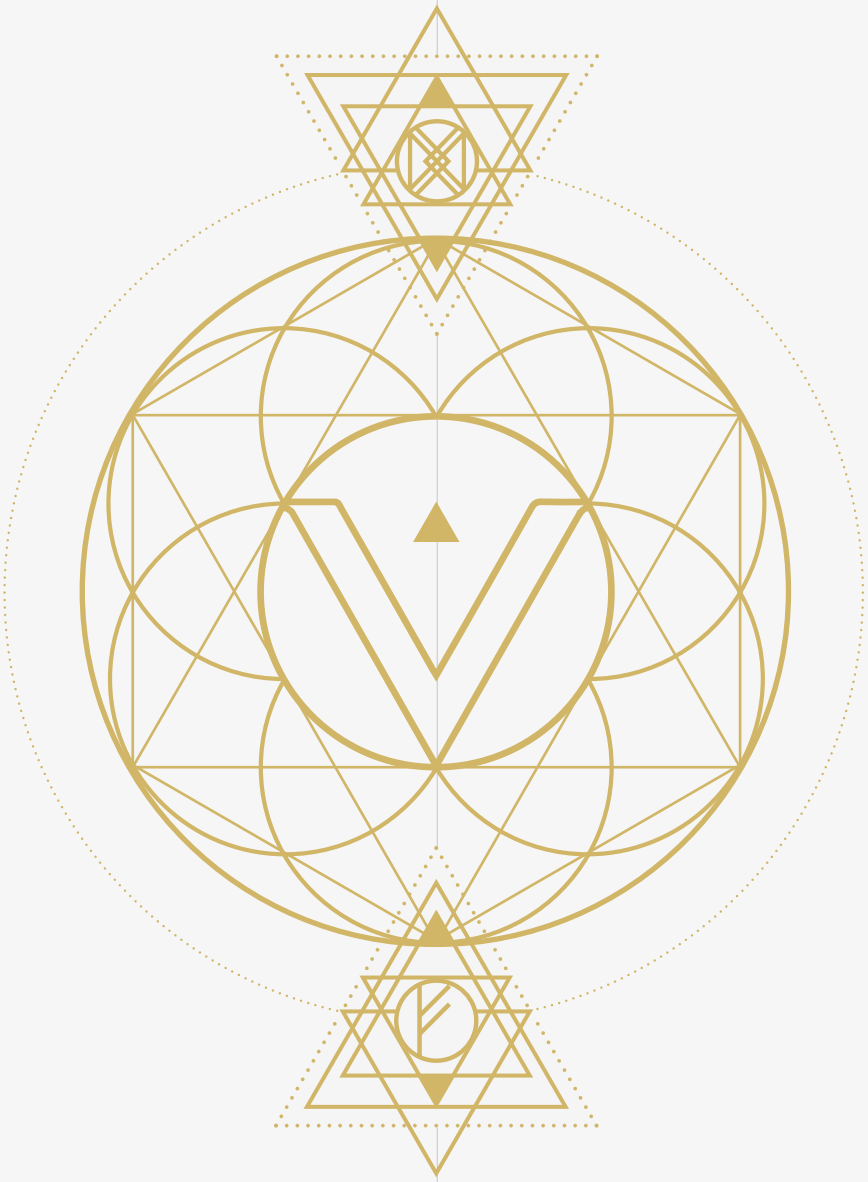
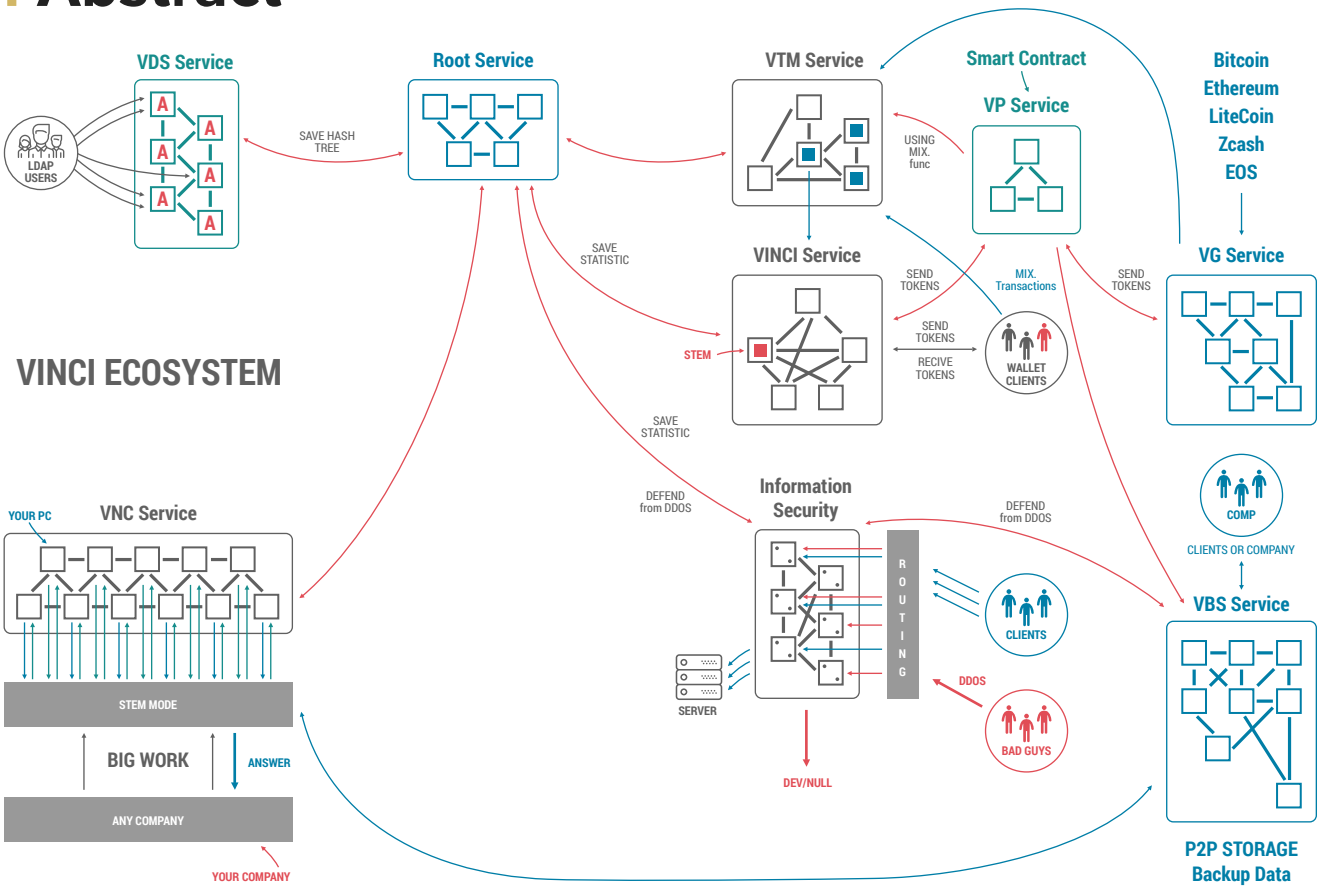


VINCI

Technology Overview

Contents:

1 Abstract



VINCI is a decentralized cloud blockchain platform that offers a pioneering architecture for building distributed internet services and computing systems using Service Oriented Architecture (SOA). Each Vinci software service is a dedicated blockchain service that any user can implement and use. All services together form a computing ecosystem with the following features: scalability, high speed, decentralization, interoperability and fault tolerance.

2 Actual requirements for blockchain platforms

Today blockchain technology develops like no other. The advent of this technology has brought along new venues for developing and improving the existing approaches to data storage and transmission. Nevertheless, progress is accompanied with new requirements being put forward by end users: businesses, governments and individuals all over the world. Platforms that fail to meet those have no future.

2.1 SERVICE-ORIENTED ARCHITECTURE (SOA)

Blockchain possesses a number of advantages appealing to businesses, specifically: tremendous computing power, security, reliability, fault tolerance, simplicity and transparency, however, none of the existing blockchain platforms offers mechanisms to implement private business logic on top of its architecture. Their functionality is limited to payments and smart contracts, the two features being narrowly applicable. This hinders the industry development significantly: we cannot use the existing blockchain platforms to build new services and transfer the existing ones into them. The need has become evident now for a new service-oriented architecture. Future systems must provide APIs for creating services running on a single computer network and offer inter-service data exchange.

2.2 EFFECTIVE SCALABILITY

User traffic and data flow on blockchain networks grow with every passing day. The throughput capacity of popularplatforms cannot catch up with the ever growing requirements. Every day sees new ideas and their adaptation algorithms: an increase in the block size, variations of consensus algorithms, the use of side-chain solutions and others. All this will only have a temporary effect, for understandable reasons. New platforms must be designed allowing for the fact that the number of users and data flow will be dozens of times bigger in the future than today. The only way out is efficient scaling. Unlike centralized systems, blockchain makes it easy owing to its decentralization. Division of the input flow and parallel data processing at numerous software services make the system capable of operating with equal efficiency for tens of thousands and tens of millions of users.

2.3 BUSINESS-ORIENTED PERFORMANCE

The total hash rate of the bitcoin platform has reached 4 Exa hashes per second. Few supercomputers can boast such performance. However, all these computing resources are expended only to maintain the operation of the obsolete consensus algorithm, PoW. Statistics show that over 90% of companies prefer to use virtual and cloud technology instead of buying their own resources. The use of the resources of blockchain networks for useful computation, in particular for implementing a service-oriented architecture, can attract many more interested users and expand tremendously blockchain application fields.

2.4 SAFETY AND LIVENESS

Decentralized systems have special requirements for their security, ability to recover, and upgrading processes. Platforms without such mechanisms can only change through forking, only to give rise to even more problems for users. It is also worthwhile noting that forks are simply unacceptable in business applications. A special approach must be developed that will enable the system to evolve not only without forks, but also without critical delays. In the absence of such mechanisms, the system will not be competitive and user-friendly.

3 Service-oriented architecture

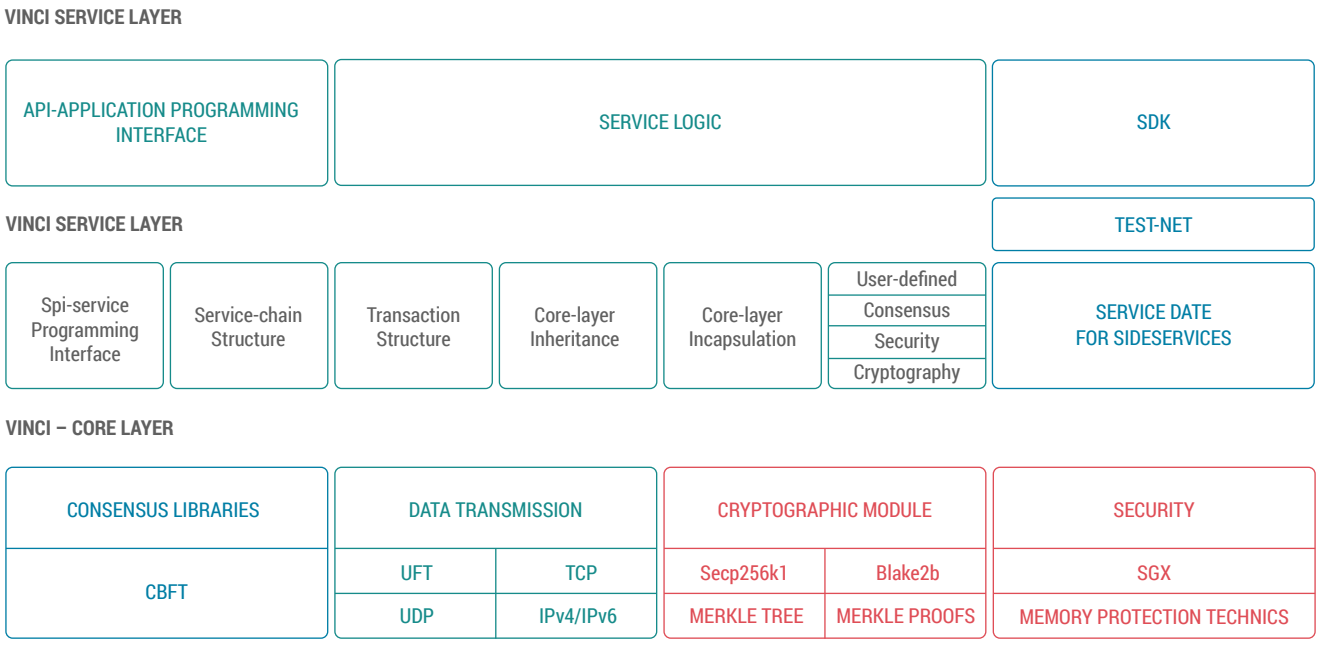
Every software service in Vinci ecosystem is a special isolated service performing regulated functions, capable of information exchange with other services within the system. In other words, VINCI is the pioneering approach to creating the architecture of decentralized internet services. Such services include DNS, DAR, LDAP, Artificial neural networks, routing, proxy, distributed computing, certifying centers, payment functions proper, and business-oriented services with free-format transactions and a block structure for implementing business logic especially needed by companies and corporations unable to maintain computing resources of their own.

There are two types of software services in VINCI: static and dynamic. A static service is a system-defined system that serves to maintain its operability and expand the functions of the core. A dynamic service is a user-defined system that can be created by any user for implementing private logic. Any dynamic service can become static if its functions are critical to the system and it receives the community's approval.

One should not think of every individual service as a separate set of physical machines that maintain its operation. Numerous services may have a purely virtual structure because they are meant to be managing or controlling entities for service working under a higher load. While static services are thought to ensure vertical scalability, dynamic ones do horizontal.

The VINCI core has no built-in mechanisms for processing smart contracts, anonymizing transactions, load balancing, and others because all these are services and they must be distinguished as individual services. This way, VINCI comes to gain advantages that other platforms lack. The system architecture is flexible and can at any moment change its operation principles by modifying and creating new services. Furthermore, VINCI imposes no restrictions: anyone can use their own types of smart contracts and their own types of virtual machines to process those contracts, create their own transaction mixers and any other services expanding the functionality of the entire system – and all these are just individual services.

4 VINCI Anatomy



4.1 DESIGN PRINCIPLES

4.1.1 KISS – KEEP IT SIMPLE, STUPID

The system performs better if it remains simple and clear. This not only lowers the entry barrier, but also enables easy modifications and improvements. The system must be transparent and easy to learn, service not complex and formidable, scaring away users and developers.

4.1.2 BE A PART OF COMMUNITY

The community of developers of open source systems provides phenomenal opportunities for active, by-the-hour system development. People united by the same idea and goals are the strongest driver of progress. Support from these people, including financial, their motivation makes it possible to create truly fundamental platforms capable of competing with closed and commercial organizations.

4.1.3 FLEXIBLE IS BETTER THAN SOLID

One way to build a competitive system that will be able to efficiently develop every day is to keep to a minimum the number of functions built into the core and provide mechanisms for implementing new functionality by means of flexible modules that can easily be loaded into and unloaded from the system.

4.1.4 KNOW YOUR USERS' REQUIREMENTS

The only right direction for developing a system is a dialogue with users. They alone know what functions are necessary today and what will be necessary tomorrow. In the absence of this communication, a system risks going the wrong way and finding itself redundant and useless.

4.1.5 SAFETY

One of the first requirements important to a user is absolute security of personal data. Although this lies at the heart of the blockchain philosophy, not many platforms under development pay due attention to it. They go the wrong way simply copying their predecessors. Only with an experienced team of developers, engineers, mathematicians, and cryptographers on board can we develop a system that will fully meet this requirement.

4.2 SOFTWARE SERVICES STRUCTURE

VINCI elements structure

TYPE	FUNCTIONS	NEEDED POWER
END CLIENT	• TRANSATIONS • CONTRACTS • VOTING	LOW
TWIG NODE	• SAVE CHAIN • COLLECT TRANSACTIONS • CHECK TRANSACTIONS • BUILD BLOCKS • RESEND TR. & BLOCKS • GATE FUNCTION • VOTING	MEDIUM / HIGH
STEM NODE	• SAVE CHAIN • SYNCHRONIZE • EXCLUDE • CHOOSE	MEDIUM / HIGH
ROOT SERVICE	• SAVE ALL CHAINS • GIVE DATA TO WORLD • ROUTING	HIGH

Every Vinci service consists of a set of nodes with various functions. A node that participates in the consensus algorithm, gathering of transactions, and forming and validation of blocks is a Twig Node. A node that participates in the CBFT consensus algorithm speeding up its operation is a Stem Node. The function of a Stem Node is exclusively auxiliary: the system can function without it, but a different consensus algorithm is used in this case. Anyone can become a Twig Node for a service by filing an application and passing selection. A Stem Node is serviced by the creator of the software service if it is necessary, for its operation, to use the quick and efficient CBFT consensus algorithm. If the data processing speed requirements are not as important, a software service can function independently.

4.2 SOFTWARE SERVICES STRUCTURE

To perform these functions, the Root software service stores in its blockchain and operates the following types of data:

1. Software service name;
2. Unique Software Service identifier;
3. Access modifiers (public / private / secure);
4. Software service activity sign;
5. The structure of Merkle Trees for every blockchain of every software service.
6. The Root software service does not store copies of all blocks: it uses a Merkle Proof for data validation;
7. A description of the service provided by a software service and its API.

The Root software service supports the following types of transactions:

1. New software service registration transaction;
2. Software service transactions;
3. Software service block Merkle Tree saving transaction;
4. The transaction of saving the Twig Nodes for a current software service validation session.

4.3 ROOT SERVICE

The main service of Vinci is called the Root. It coordinates the operation of other Software Services in eco-system when they exchange data. The co-ordination is based on the following functions:

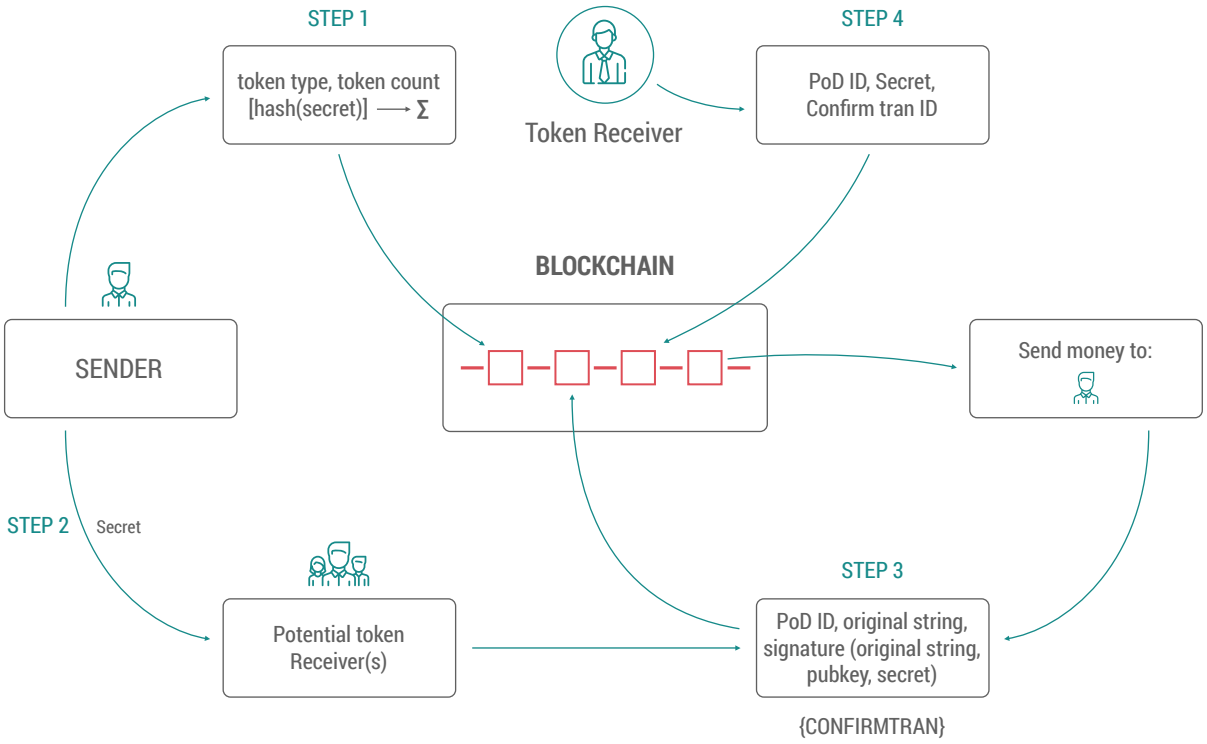
1. Confirmation of the validity of data received from another software service;
2. Provision of a list of active services;
3. Provision of a list of active Twig Nodes of a software service;
4. Provision of service data about the software service.

4.4 VINCI SOFTWARE SERVICE

The VINCI service provides the service of work with the main coin of the system, VINCI. This is the function of transfer of funds between system users and support for a number of special transactions:

1. A transaction for filing an application for participating in a current validation session as a Twig node;
2. On-demand transactions (PoD);
3. A transaction for transferring funds;
4. A transaction for transferring funds with Multi Signature support.

4.5 VINCI PoD – PAYMENT ON DEMAND



This type of transaction is implemented in the VINCI Service and provides the functionality of transfer of funds between users with the participation repudiation feature. A user can switch any number of coins to a special “deposited” state from which they can be withdrawn only after a secret is presented. The secret may be presented by any network participant because coins in a “deposited” state are linked neither to a sender nor to a possible recipient of funds. The transaction creator can generate the necessary number of secrets to divide the amount between them in the necessary proportions.

4.6 APPLICATION SOFTWARE SERVICE OF VINCI ECOSYSTEM

The ideology of SOA suggests eco-system of software services. Every service in VINCI has it’s own function and these words are considered interchangeable.

There are two ways to build services in eco-system:

The first is using the structure of blockchain services storing key-value records that define the functionality of the service.

The second approach is more comprehensive and offers more resource-intensive services. They need computations that will define the contains of the blockchain (smart contracts) or the blockchain will carry out solely administrative functions when the output data is not intended to be stored on blockchain (rendering, scientific researches, DApps).

In both cases each service must provide its API description upon entering the system. When creating a software service, it is possible to create a token associated with it. In doing so, you must specify its generation rule with the software Service working upon registering on Root Service.

4.7 SOFTWARE SERVICE FORKS

Vinci does not use forks. For a regular blockchain, a fork is a change of the built-in operation algorithms that are critical to the entire system. Thanks to its modular structure, Vinci enables expanding its functionality by creating new service. The end user is free to choose the services they trust, so simultaneous existence of two or more services that have the same functionality but different implementation (for example, transaction mixers) is the standard mode of Vinci operation.

4.8 VINCI TESTNET

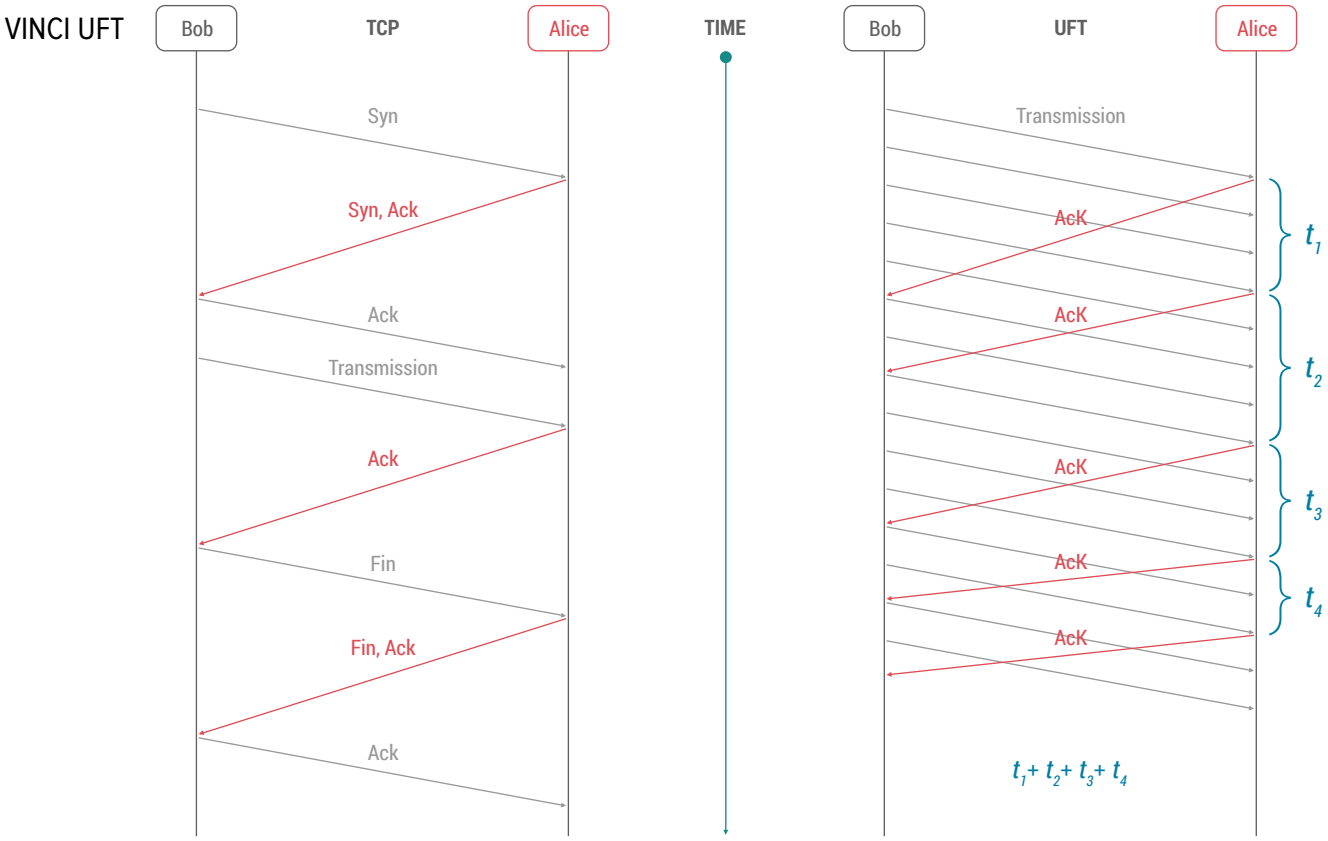
The Vinci testnet consists of two elements. The first is a purely virtual network emulating the operation of the Vinci network on a single computer. The second is a live testnet work created and supported by the Vinci Technologies for full-scale testing of new software services, especially when candidates for static software service appear.

4.9 PARALLEL TRANSACTIONS EXECUTION

The high speed of the operation of Vinci implies special requirements for algorithms processing incoming transactions. To avoid queues, parallel input flow processing is used. Transactions are distributed between parallel service flows depending on their type and a special value of the flow label obtained from the sender's address. When a validator is triggered, the software determines the possible number of system flows that can be used to process transactions. Depending on that value, an appropriate number of processing pools with unique identifiers is formed to which incoming transactions will be directed. There can be several options for obtaining a flow label from the sender's address. The simplest is division of the address space by the number of flows. Compared with sequential processing, this yields a significant increase in performance, but continues to protect against double spending.

4.10 UFT – UDP FLOW TRANSMISSION

The UFT (UDP Flow Transmission) protocol processing the transmission and retransmission of data between software services participants (and directly between software services) helps to avoid problems typical for TCP (which is the de facto standard for data transmission on the Internet): slower transmission speed due to packet loss and congestion (when the client cannot receive all data the sender has sent). Each of these problems leads to a situation when an operation such as sending transactions for validation may significantly decrease the network speed because of utilizing the network resources of each node for additional, in the case of blockchain, unnecessary overhead costs.



A decline in the transmission speed in the case of TCP occurs because of possible congestion of backbone routers when the message queue is congested. As the number of routers in the network grows, the loss probability of increases, which even in the case of a single loss leads to a quite noticeable decline in the transmission speed. The TCP protocol suggests mutual confirmation of each packet in the transmitted flow, so if the sender does not receive the acknowledgement signal (TCP ACK), these leads to re-initialization of the transmission from the point where the packet was lost.

UFT suggests establishment of a two-sided connection between sender and receiver where the sender-receiver connection is used for the sent data flow and the reverse, independent, one is necessary for receipt acknowledgement. The fundamental difference between TCP and UFT is that UFT requires selective acknowledgement of the received flow (SACK), after short time intervals.

Periodic ACKs help simplify managing the backward movement of control messages when the data transmission speed is high because in these situations the number of ACKs is proportional to time, not the number of data packets as in the case of TCP. If a packet is lost, the sender sends a negative response (negative ACK, NAK), which leads to selective resending of a specific part of the sent flow.
One of the congestion control strategies used by TCP is the so-called “Slow start and additive increase” which makes it impossible to send small packets containing transactions in order to use maximum network throughput. When transmission begins using the TCP protocol, the speed is set at the minimum with further increase as the process goes, but this approach totally fails to achieve its goal because transmission ends at the moment when its speed has not yet reached the maximum. UFT, on the other hand, uses a channel width control mechanism based on decreasing the data transmission bandwidth proceeding from optimum values sent by the ACK and NAK sender.

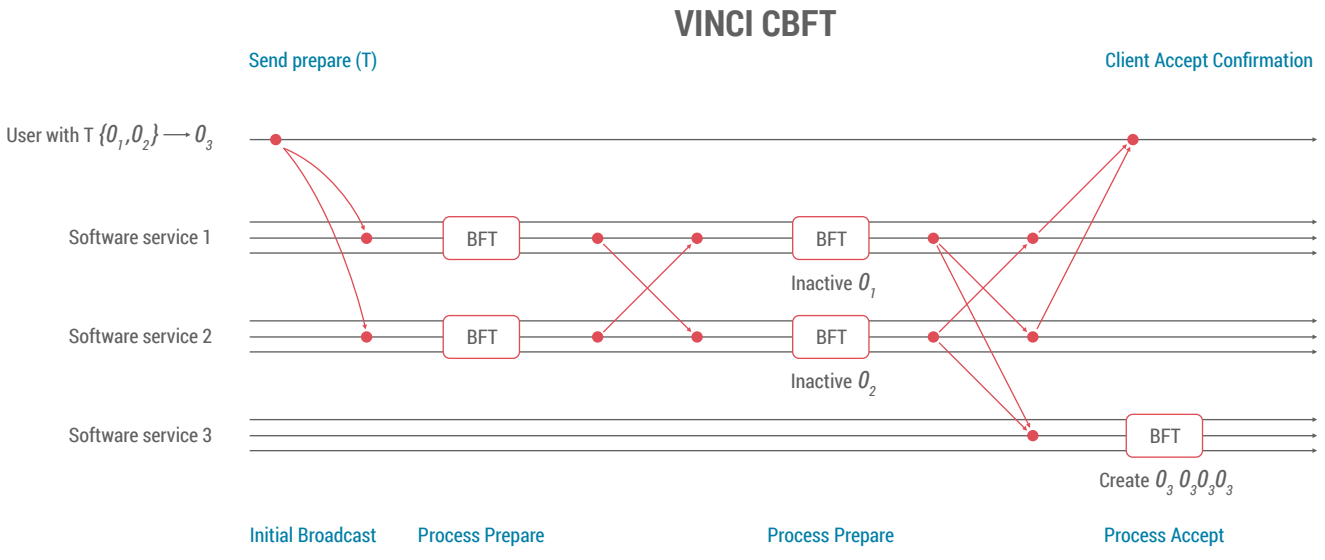
Thus, replacing standard data transmission protocol with UFT provides maximum efficiency in terms of network and with usage, which leads to faster exchange of information between the nodes and faster block validation, and this has a significant effect on the total number of processed transactions and, first of all, on the speed of exchange between the software services.

4.11 CONSENSUS

4.11.1 CBFT consensus

Convolutional Byzantine Fault Tolerance is the main consensus algorithm in Vinci. Unlike popular PoS, CBFT does not divide system nodes according to the size of a certain contribution. They are all equaland there is just a set of criteria used for their selection:

– computing power;
– internet connection quality;
– having enough coins for the deposit.



The system uses the deposit to guarantee the trustworthiness of a node. Nodes are randomly selectedat fixed intervals from among those having filed an application. To file an application, a special type of transaction is used for transparency of the process, and data about all candidates and Twig Nodes are stored in the blockchain. The number of Twig Nodes operating simultaneously during a current validation session depends on network load. The duration of a validation session is 10,000 blocks.

A distinctive feature of the CBFT implementation is the absence of the need for sending out a candidate block to the entire network of Twig Nodes. Instead, to reach a consensus and hold a vote it is enough for nodes to exchange a compressed set of metadata about transactions. Sending of block is only used when new validators appear that need a current copy of the blockchain. In addition, the use of the UFT protocol enables avoiding forced delays in transmission the way it happens in the classical implementations of the BFT algorithm. The beginning of the process of the formation of a current block is controlled by a Stem Node that randomly selects a current Twig Node from among active Twig Nodes. A block takes 1.5 seconds to form, and only one validator has the right to form a block at a given moment of time. If no block is formed after an interval of 1.5 seconds, the Stem Node transfers control to a next Twig Node and doubles the interval. If a validator fails to form a block over an allotted period of time more than 3 times in a row, it is removed from the current set of validators. If a block is formed with false transactions, the validator is deleted and the system withholds their deposit.

4.11.2 CBFT Technical

The network architecture of the consensus systems of existing blockchain platforms is based on the principles that the system has a stationary operation period and is built entirely according to the Poiss on point process. The need to analyze the form of the input data flow distribution and the duration of transition periods is in many cases determined by the fact that the latter may make up a significant proportion of the system operation period, while the input flow distribution may have a significant influence on the statistical characteristics of the output parameters of the system. Therefore, without taking account of the non-stationary period and the influence of the form of the input flow distribution law, it is impossible to optimize the operating characteristics of the system as a whole.

Consider a model of the input flow of jobs in the system that is determined by dependence of the input flow rate on time $\lambda_g(t)\ (g=\overline{1, period})$, given the average rate λ , i.e. the input data are comprised of the value of the average input flow rate λ and statistical information reflecting change in the rate of the input flow of jobs over a certain period of time (period – the system operating period). Queueing theory generally defines that jobs arrive in the system exponentially and the rate of the input flow of jobs λ, does not depend on time t. The time interval between incoming jobs ξ is a continuous random value having exponential distribution with the parameter $\lambda > 0$. ξ has only non-negative values, and its density $f(x)$ and distribution function $f(x)$ are as follows, respectively:

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

$$F_\xi(x) = \begin{cases} 0, x \leq 0, \\ 1 - e^{-\lambda x}, x < 0; \end{cases}$$

Where λ is the input flow rate.

The mathematical expectation of the random value ξ is connected with the input flow rate by the $M_\xi = \frac{1}{\lambda}$. The dispersion of the random value ξ is also determined by the input flow rate $D_\xi = \frac{1}{\lambda^2}$.

In practice, the system input flow rate is not a constant value, it changes in time. These changes are connected with the period of sending out a candidate block as consensus is being reached between active system participants and with the service time in the system, namely with the analysis of the candidate block which consists of a set of transactions for the current validation period.

As the CBFT consensus algorithm was being developed, statistical information was collected reflecting change of the input data flow rate in the distributed network depending on the time it took to distribute a candidate block, its size, and processing time. The approximation of the obtained statistics is carried out with the help of the piecewise-polynomial interpolating cubic spline $S_g(t)$ with the setting of boundary conditions, i.e. for each section $[t_j, t_{j+1}]$ with the number j the approximation function $S_g(t)$ has the formof a polynomial.

$$P_j(t) = \sum_{i=0}^{k-1} a_i^{(j)} (t - t_j)^i, k - 1 = 3$$

The boundary conditions consist in the periodicity condition, i.e. coincidence of the values of the first and second derivatives on the borders of the interval $[T_1, T_n]$. The plotting of the spline can be reduced to determining the multiple ratios $a_i^{(j)}$ by solving systems of linear equations. The interpolating spline $S_g(t)$ is plotted so as to satisfy the interpolation condition $S_g(t_i) = y_g(t_i)\ i = 1, ..., N$ for the table function Y_g.

To obtain the specific dependence of the input flow rate on time $_g(t)$, given the average rate λ , it is necessary to multiply the obtained spline $S_g(t)$ by the average rate and divide it by the average value of the spline $S_g(t)$ itself:

$$\lambda_g(t) = \frac{\lambda}{\overline{S_g(t)}} S_g(t)$$

$$\overline{S_g}(t) = \frac{\sum_{i=1}^{N} S_g(t_i)}{N}$$

where λ is the actual average input flow rate.

The proposed input flow model suggests that the time between incoming jobs ξ is a continuous random value that may be distributed according to different distribution laws, such as the exponential law, Poisson law, normal law, and uniform law.

4.12 ATTACKS PROTECTION

Blockchain, like any internet-based information system, is a target for hackers. Besides the known type of attack, DDoS, new methods appear for affecting specifically blockchain platforms. Analysis of all these threats, includingresearch into the shortcomings of the architecture of existing blockchain systems that have been subjected to attacks, has made it possible to build effective methods for proactive defense into the Vinci subsystems.

4.12.1 DDOS – DISTRIBUTED DENIAL OF SERVICE

The success of this attack is based on limiting throughput which is one of the characteristics of any network-based resource. A large number of requests are sent to a resource under DDoS attack with the aim of exhausting its potential for data processing and disrupting its normal operation.One of its tasks is constant monitoring and analysis of data, so when there are signs of a DDoS attack, the traffic is routed to special nodes that imitate processing. As a result, the attacker gets the impression of a successful attack, while the protected service continues normal operation.

4.12.2 SYBIL ATTACK

During a Sybil attack, the perpetrator fills the network with numerous controlled nodes trying to “surround” the victim’s node, i.e. get control of all neighboring network nodes. This type of attack is not viable when the CBFT consensus algorithm is used because any attempt at disrupting normaloperation of the network will result in blocking the node and the system withholding its deposit. Even if the perpetrator has enough resources to continue the attack, nodes are selected randomly for everyround of validation, which significantly decreases the probability of success.

4.12.3 ECLIPSE ATTACK

This kind of cyber-attack was described in detail in a 2015 report by a group of scientists from the Boston and Hebrew universities led by Ethan Heilman and later demonstrated using the Ethereum network as an example. Using correct manipulation in a peer-to-peer network, a hacker can “obscure” nodes in such a way as to make them contact only infected nodes and thereby influence the process of forming new blocks. For the consensus algorithms used in Vinci, this attack is unrealizable. CBFT, due to random change of validators in every validation round, forces the attacker to start the attack afresh every time, which negates its effectiveness.

4.12.4 51% ATTACK

This attack is among the most popular and has been carried out against mining-based blockchains many times. For other systems, not based on mining, including for Vinci, this attack becomes difficult to carry out because it requires orders of magnitude more economic resources. If the SGX-CBFT hybrid is used, the attack becomes altogether impossible because even if one node is captured, it will instantly be localized and blocked thanks to the possibility of remote reliability verification.

4.12.5 DOUBLE-SPENDING ATTACK

This attack suggests that the same funds are successfully used more than once if blockchain goes out of sync. This attack is only typical of systems in which every network participant has the right to form new blocks. For the CBFT consensus algorithm this attack is unrealizable because only one node is responsible for forming a block over a specific period of time and the network cannot go out of sync.

4.13 SWIFT TORUS ROUTING

Modern internet routing protocols were created to be used in centralized systems. Existing decentralized platforms use them because there are no analogues and alternatives. These protocols are, nonetheless, not an efficient solution to the problems of routing in decentralized systems because they operate absolutely different entities and the need for fast routing between dynamic nodes is much higher.

Decentralized networks include a large number of transit nodes, orders of magnitude more than centralized. Solving the problem of routing on such networks not only requires mathematics that is more complex, but the very calculation of routes becomes impossible without creating special software. To solve these problems, a protocol called Swift Torus Routing is being developed. Its use enables considerably increasing the efficiency and speed of data transmission between dynamic nodes, namely when transmitting data between nodes of different software services in the absence of a centralized point of entry into the system implemented by a software service.

One method for building dynamic routes in systems with a dynamic network topology is the use of graphs whose points are network nodes and connections between them are weighted communication channels. Instead of standard graph traversing algorithms, the calculation algorithm used to find the shortest path is based on space-efficient graph representation – Hilbert-ordered tiles. This method is used in a modern graph processing engine named MOSAIC. For comparison, 4 Xeon Phi processors using the Hilbert-ordered tiling scheme take one second to calculate routes based on a graph of approximately 700,000,000 points.

Discrete time models are used to form the network graph proper. These types of models represent network topology changes as the periodically recurring sequence S of topology snapshots divided by the duration interval $\Delta t=T/S$, where T is the recurrence period of the topological states of the set.

Each snapshot is matched to the graph G = (V, E) where V is the set of nodes and E the set of communication channels. For the finite set of graphs {G} recurring over the period T, routing table are pre-calculated. These tables are distributed between nodes and used when necessary Δt .

The mathematical model of the system makes it possible to present continuous topological changes in the form of a sequence of steady states described with the help of graphs:

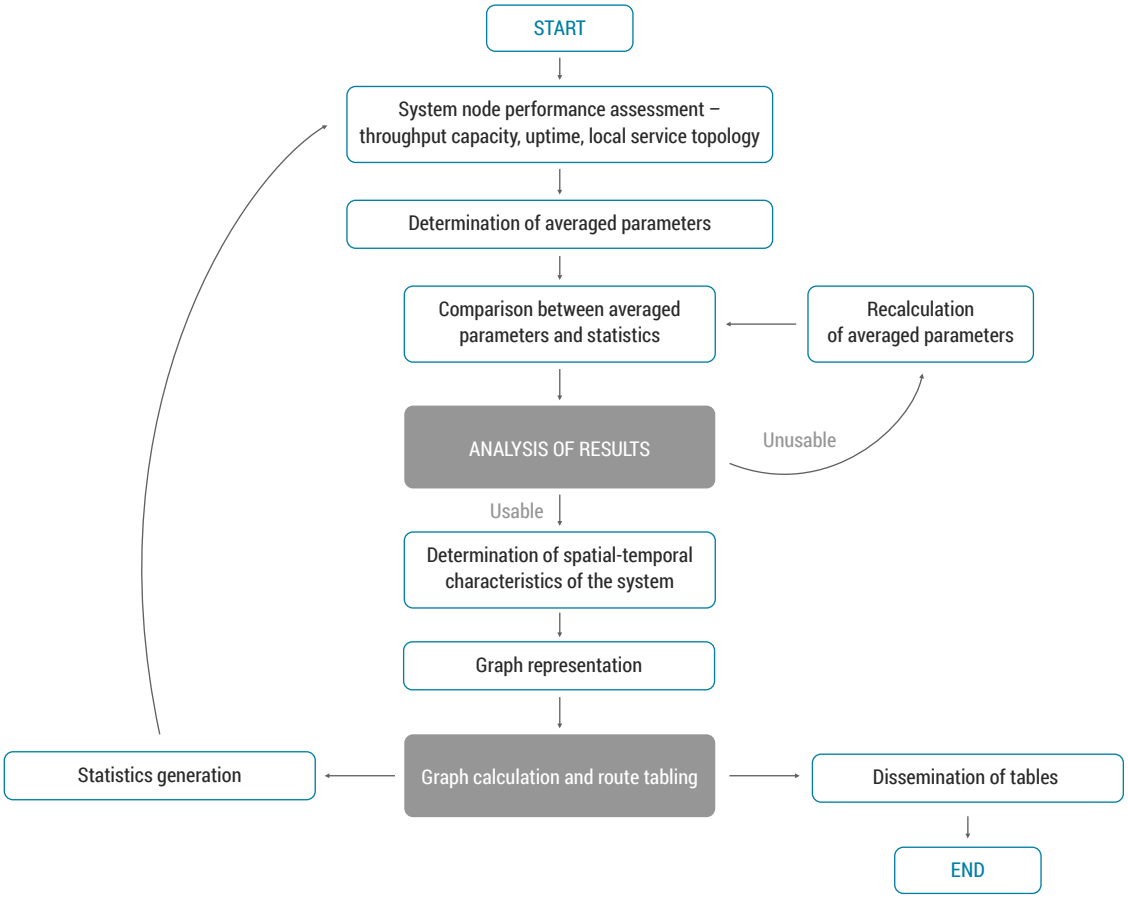
$$\forall t \in [0; +\infty) \exists \Delta t_n = t_n - t_{n-1}$$
$$t \in (t_{n-1}; t_n)$$
$$G_n(t) = G(\Delta t_n)$$
$$G(\Delta t_0) \rightarrow G(\Delta t_1) \rightarrow \ldots \rightarrow G(\Delta t_n)$$

to take into account that the topological state of the network changes in time under the impact of external events:

to forecast the future state of the network topology for a certain time – the forecast time.

$$\sum_{1}^{n} \Delta t_n = T_p$$

where Δt is the time during which the system parameters are considered not to worsen; $G(\Delta t_n)$ is the network graph at the moment with a finite set of points V and a set of edges E. T_p is the time of the forecast state of the communication system. Considering the above-stated criteria, the key input parameters of the model are statistics on the operation of nodes and channels and performance characteristics of equipment. Below is a flow chart of the algorithm of pre-calculation of routing tables.

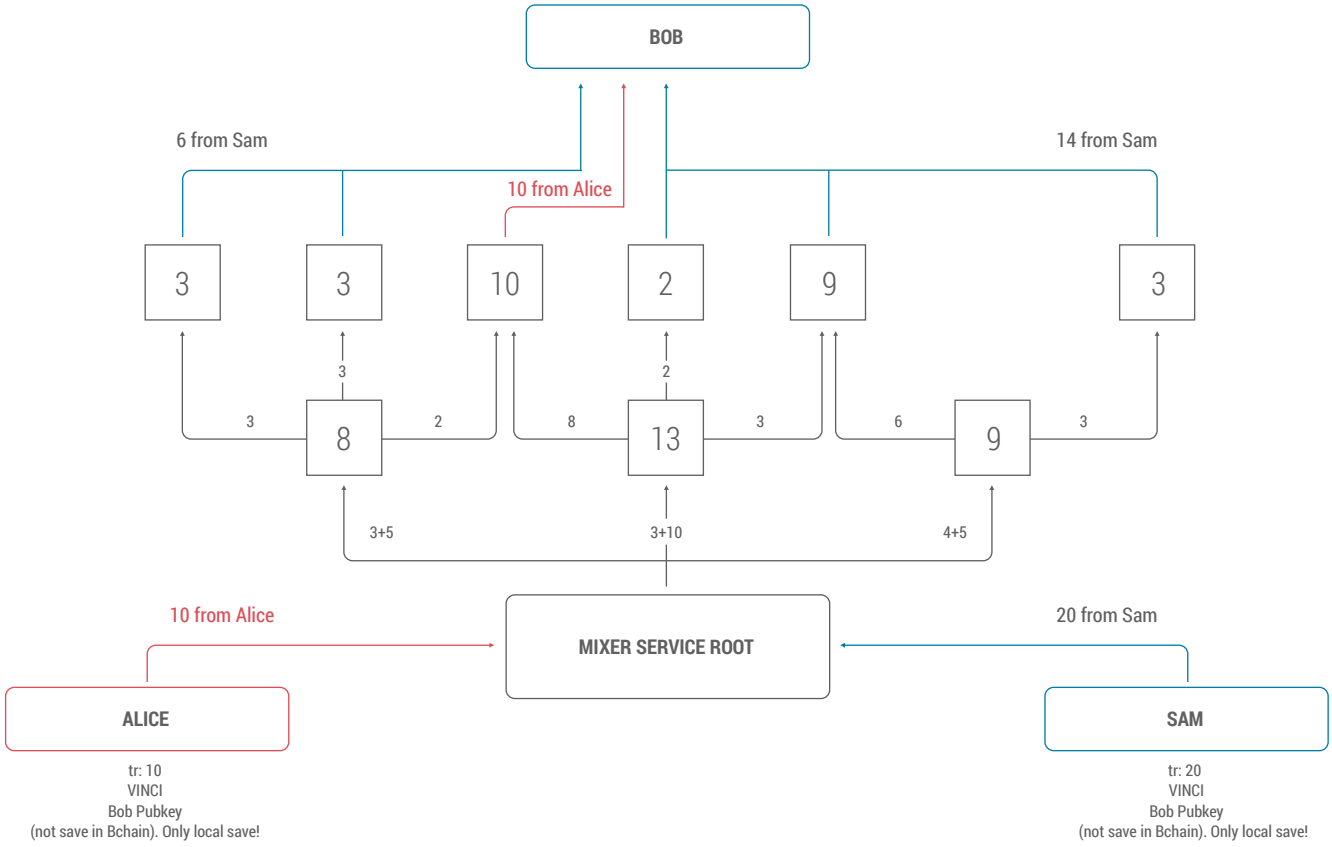


4.14 VINCI SERVICES APPLICATIONS

The Vinci Technologies community of developers will create the main business-critical services based on the principles of decentralization and security. Their task is to serve as the basis for the operation of services of greater complexity that stakeholders will develop.

4.14.1 VTM (Vinci Transaction Mixer)

VINCI VTM

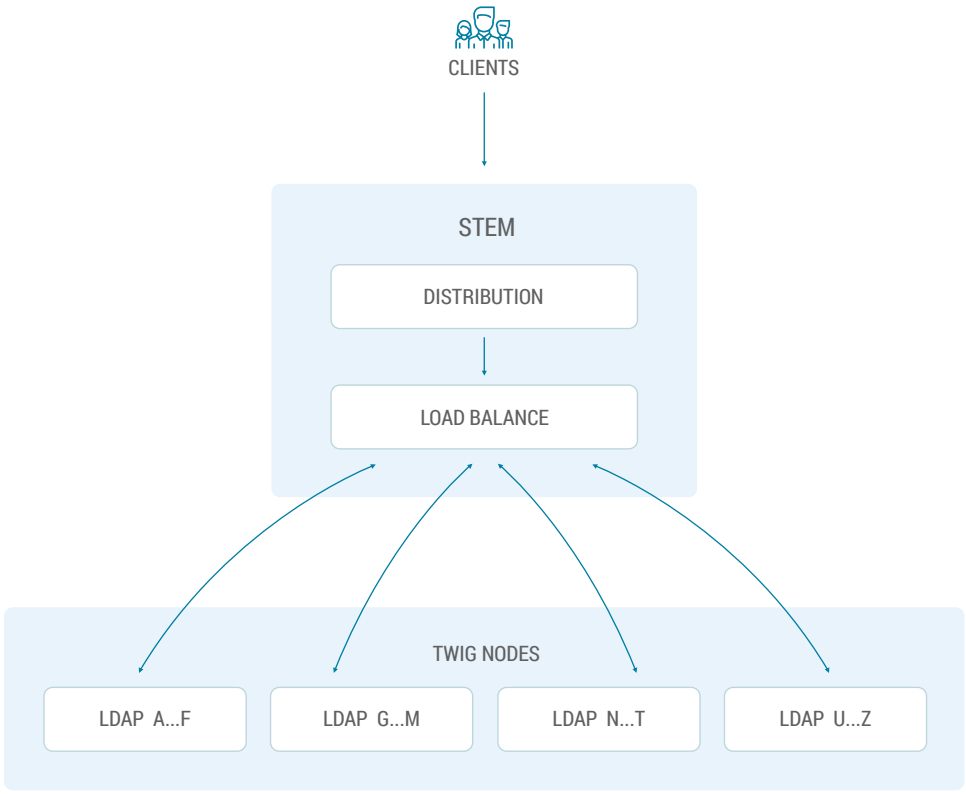


The transaction mixer, popular among blockchain users for increasing the anonymity of transactions, can be easily implemented in Vinci. A user transfers tokens to the Stem Node of a software service stating the end recipient of the funds. The Stem Node divides the amount into numerous smaller fractions and sends them to Twig Nodes with an end recipient label. Network nodes exchange these tokens during a user-selected time and send the funds to the recipient at random moments. The software service Stem Node maintains control to ensure that all funds are transferred. The software service will store no information about senders and recipients. Because several mixer options can be implemented simultaneously by different users, it is easy to use these mixers one after another, which greatly improves the anonymity of transfers keeping users' personal data secret.

4.14.2 VDAP (VINCI DIRECTORY ACCESS PROTOCOL)

VDAP is a protocol providing the functionality of the LDAP protocol with higher requirements for security, fault tolerance, and operating speed.
All stored catalogues are distributed between the nodes of this software service, and the reservation degree of each type of data can be set separately. The Stem Node determines the software service node to which a user request will be redirected, allowing for the current load on the software service and the list of the requested information.
If one or several nodes malfunction, all requests will be redirected to other nodes, transparently for the user.

VINCI DDAP



4.14.3 VBS (VINCI BACKUP SERVICE)

Using a blockchain to store distributed data is way more advantageous than using a centralized system. The speed of requesting and downloading data is orders of magnitude higher because they data is downloaded simultaneously from many sources sharing the file. The only bottleneck is data integrity and security if a distributed network participant is out of order or compromised. This drawback is easily overcome with the use of the Forward Error Correction (FEC) schemes based on the Vandermonde matrix. The Vinci Backup Service provides erasure recovery and error correction features that enable maintaining data integrity and security even if a part of them is lost. There are a number of error correction codes that can be used in FEC. These are Reed-Solomon code, Hamming code, Reed-Muller code, binary Golay code and others.

4.14.4 VNC (Vinci Network Computing)

One method for implementing distributed computing is Grid systems. A grid is a geographically distributed computing platform consisting of heterogeneous nodes accessed via a single interface. It co-ordinates the use of resources when there is no centralized management of these resources; uses standard, open, universal protocols and interfaces; and nontrivially ensures high-quality service. Grid systems are divide into several types of resources:

– Data grid;
– Information grid;
– Computing grid.

Grid applications include:
– Complex modeling on remote supercomputers;
– Joint visualization of very large sets of scientific data;
– Distributed processing for data analysis;
– Video rendering.

Leading grid solutions are based on the Open Grid Service Architecture (OGSA). OGSA is a distributed computation and interaction architecture based on services and guaranteeing interoperability of heterogeneous systems regardless of their implementation, location, and platforms in such a manner as to enable different types of resources to communicate and exchange information.

Vinci develops a VNC service based on OGSA with the aim of efficient use of computing resources both in existing projects like HTCondor, Globus Project and BOINC and in new ones, based on OGSA.

VNC includes:

– The client part of the distributed application, serving as an integrated portal to all distributed grid computing platforms;
– An SDK based on HTCondor for performing computing tasks;
– A control point mechanism;
– A data replication service.
– VNC will make it possible both to rent one's computing resources and to use the computing resources of grid platforms for performing one's one tasks.

4.14.5 VP (VINCI PYTHON)

Vinci imposes no restrictions on features to be implemented in its software service. This means that no restrictions are imposed on the types of virtual machines to be used in software service for implementing the functionality of smart contracts and DApps. The Vinci Technologies implements a new type of virtual machine for processing smart contracts that is based on a Python-like language. Python programming language has been chosen as the basis because Github statistics show that it ranks first in popularity among programming languages for quick and efficient creation of scripts which in our case are smart contracts. This software service is not the main unit for processing smart contracts in Vinci, but only provides this functionality to everyone, along with other similar software service.

5 Conclusion

The Vinci platform was designed based on experience of comprehensive research into advanced blockchain technologies, cryptography and cybernetics. Thanks to its service-oriented approach, Vinci creates the first real blockchain ecosystem to become a new step in the development of information technology and to improve the life of everyone on Earth.

6 Disclaimer

Please read the following notification properly. This notice applies to all persons who read this document. Please note this notification may be changed or updated.
The purpose of this Vinci Technology Overview is to provide the persons, pursuant to a confidentiality deed (the "Recipients" and each a "Recipient") with general information concerning the business of Vinci Technologies Limited (collectively referred to as "Vinci Technologies") and to assist the Recipient in deciding whether to acquire investment referred to in this Section as the "Transaction". It is supplied to the Recipient on that basis and it is not to be used for any other purpose, and is subject to strict confidentiality obligations.

This Information is supplied subject to the confidentiality restrictions contained in a confidentiality deed entered into between the Recipient and Vinci Technologies ("Confidentiality Deed"). The Recipient acknowledges that the terms of the Confidentiality Deed are incorporated into and form part of these conditions.

The information contained in this Technology Overview ("Information") has been prepared in good faith by Vinci Technologies and its advisers. This Information is intended as a guide only and does not purport to contain all information the Recipient may require in an investigation of the Transaction. None of Vinci Technologies, their directors, their shareholders nor their respective subsidiaries, jointly owned entities or associated companies or their respective businesses, shareholders, directors, partners, officers, employees, agents or advisers (collectively, the "Vinci Technologies") make any representation or warranty, express or implied, as to the accuracy, reliability or completeness of the Information or subsequently provided to the Recipient or its advisers by any of the Vinci Technologies Affiliates including, without limitation, any historical financial information, any estimates and projections and any other financial information derived there from, and nothing contained in this Information is, or shall be relied upon as, a promise or representation, whether as to the past or the future.

Any projections as to future events or other forward looking statements contained in this presentation represent best estimates only. These forward looking statements and the operations and performance of Vinci Technologies are subject to risks, uncertainties and assumptions that could cause actual events or results to differ materially from the estimates or expectations expressed or implied by such statements. The Recipient acknowledges that no representation is made that any forecast or projection as to future events will be achieved and the Recipient should make its own independent review of the relevant assumptions upon which the forecasts and projections are based. Further, such statements are made as at the date of this presentation. The Recipient should not rely on any projections as to future events or other forward looking statements as a statement, warranty or representation of fact but should satisfy itself as to its correctness by such independent investigation, analysis or due diligence as it or its advisers think fit.

To the fullest extent permitted by law, Vinci Technologies and the Vinci Technologies Affiliates:
– do not accept responsibility for any interpretation that the Recipient or any other person may place on the Information or for any opinion or conclusion that the Recipient or any other person may form as a result of examining the Information;
– do not accept any liability, whether direct or indirect or consequential, for any loss, damage, cost, expense, outgoing, interest, loss of profits or loss of any kind ("Losses") suffered or incurred by any person (whether foreseeable or not) as a result of or by reason of or in connection with the provision or use of the Information, or of the Recipient or its representatives or advisers acting on or relying on any Information; and,
– disclaim any responsibility for the accuracy and completeness of any projections as to future events or other forward looking statements and any responsibility to update any of them after the date of this presentation.

The Recipient acknowledges and agrees that:
– the Information may not be relied on in any way by the Recipient or any other person in assuming any contractual or other obligations or liability;
– the Recipient is responsible for any interpretation, opinion or conclusion it forms as a result of examining the Information;
– the Recipient will rely entirely upon its own assessment and advice in relation to the business, assets, financial position and affairs in which any person would be or become directly or indirectly interested in relation to the Transaction and in assuming any obligations or liabilities in relation to the Transaction; and,
– any opinions expressed in the Information are based on the knowledge and approach of the persons forming the opinion at the date that the opinion was formed and may have ceased or may in the future cease to be appropriate in the light of subsequent knowledge or attitudes.

Vinci Technologies may in its absolute discretion, but without being under any obligation to do so, correct, update or supplement this Information. Any further information will be provided subject to these same Vinci Technology Overview.

Questions, comments and requests regarding this Technology Overview are welcomed and should be addressed to info@vinci.id