Technical Dossier v1.1

June 2018

RADJAV

http://www.radjav.com/

FogChain, Corp

radjav@fogchaininc.com

All content in this dossier is subject to change, additionally all content is available for community debate and change.

Table of Contents

? Note

This document is the accompanying document for the RadJav whitepaper. To better understand this document and what we're trying to accomplish, please read the RadJav whitepaper first.

RadJav is essentially taking existing open source libraries and creating a JavaScript wrapper around them, making a lot of the work much easier to perform.

? GUI

- The desktop version of RadJav utilizes wxWidgets to handle the GUI.
- The HTML5 version uses Dojo.
- In an upcoming version, the HTML5 version will switch from Dojo to our own custom DOM objects as soon as possible.
- Desktop theme support is coming soon, while HTML5 theme support is already supported.
- A library similar to jQuery may or may not be created which would allow developers to port their web apps easier.
- A responsive GUI OpenGL framework would be created that would allow for responsive apps to be created using the same code. This framework would allow for new never seen before UI's to be created, which would be a UI designers dream.
- Depending on community response, we may simply make the GUI render in a webview across all platforms, similar to React Native, Electron, PhoneGap, etc. We would prefer to not do this because we feel that HTML/CSS is an ancient technology and we need to create something new. Additionally, RadJav even in it's current state can already render a GUI across all platforms in a webview,

and will continue to do this. It’s something to consider, and we are open to debates about this.

RadJav Interact

- RadJav Interact will be a Model-View-Controller (MVC) framework that will give developers the ability to create UI apps quickly and securely.
- Will have a built-in web socket server that also has a light database built-in. This will be called the Interact server.
- The UI created with the framework could connect to the Interact server and sync UI elements with it as data changes.

Game Development Tools

- The desktop version utilizes Ogre 3D
- The HTML5 version uses Three.js.
- A physics engine will be implemented.
    - Currently looking at using Open Dynamics Engine for desktop.
    - Possibly may use Physijs for HTML5.
- World building and programming tools will be created as well, community planning will go into this at a later date.

Server Apps

- Will use Civetweb for HTTP/HTTPS web servers and websocket server.

⍰ Blockchain V2 (XRJV2)

Goals

- Create a competitive and extremely fast network.
- Limit bloating as much as possible.
- Create a network that can defend itself.
    - Automatically remove harmful nodes and smart contracts that harm the network.
    - The network can defend itself through legal action, from DDOS attacks, and security vulnerabilities.
- Simple fee structure that benefits both nodes and developers.
- Never have to fork, the network itself decides on the upgrades.

Introduction

We are introducing Proof-of-Competition (PoComp) that allows for the creation of an extremely scalable and high performance blockchain. Nodes on the network would be placed into teams, with each team containing its own blockchain. The nodes on each team would be competing against each other to earn the position as team captain. Coins generated from every block would be distributed based on a node’s performance. Any node that has been promoted to team captain will get the most coins on the team.

Please note that upon launch of V2, not all features listed in this document will be available. We will work with the community as to what the most important features of V2 are, and launch those features. We will try to include as many of these features as possible for the launch.

Network Design

The biggest issues currently facing blockchain technologies is their ability to scale and process data quickly. By splitting the network into teams, we can have the network scale upwards automatically as more transactions and smart contracts are submitted to the network. Additionally, with this design we can have the network scale down as there's less traffic.

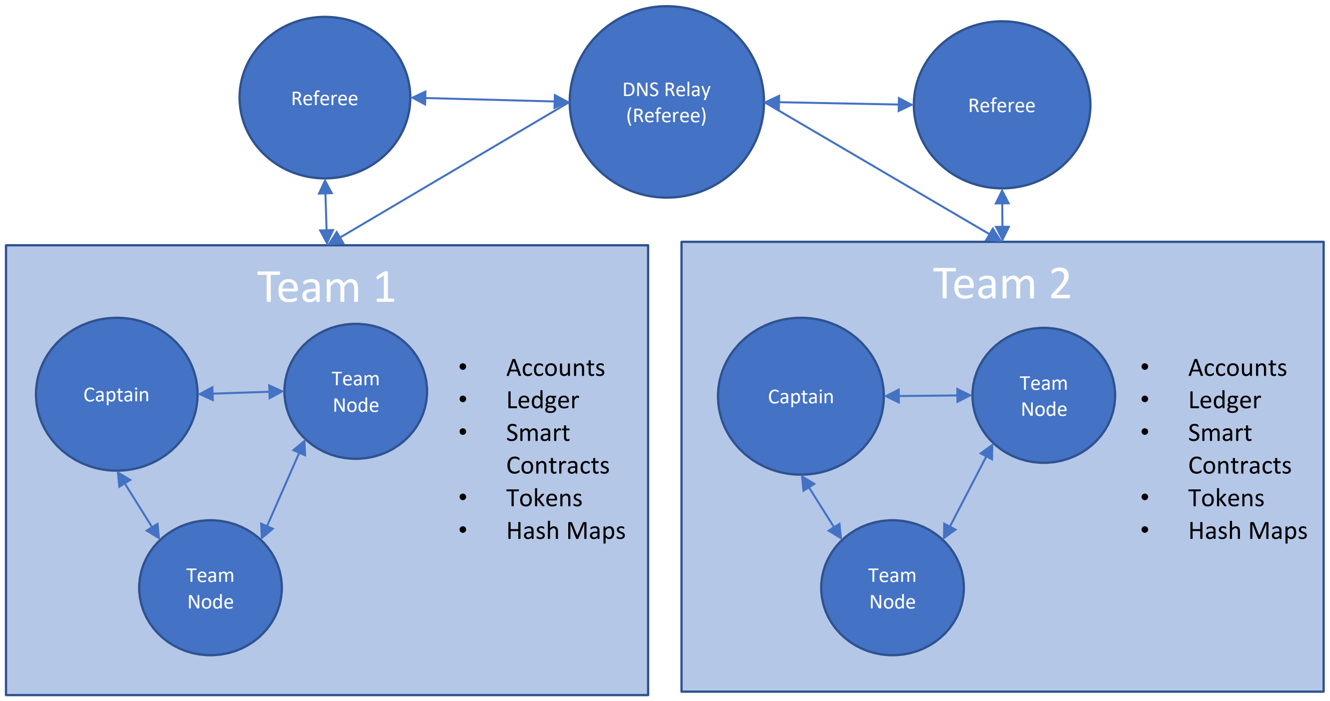
- Every team has its own blockchain, which would include the accounts and ledger.
- Each team would have a shared database which would contain the smart contracts, tokens, and hash maps.
- When a team starts to become overwhelmed with traffic, new nodes are recruited from other teams, or new teams are created.
- As a team begins to see less traffic, their nodes are offered for recruitment to other teams.

Team Design

Every team has 4 different types of nodes.

- Team node
    - Regular node, processes data, transactions, and smart contracts.
- Captain
    - Generates the block hash, does everything a team node does.
- Referee
    - Ensures that all data being processed by every node on the team is correct, and locks any smart contract that has been proven to be malicious by a network selected cyber security organization.
- DNS Relay
    - Directs incoming traffic to the correct team or node.
    - Must be a referee.

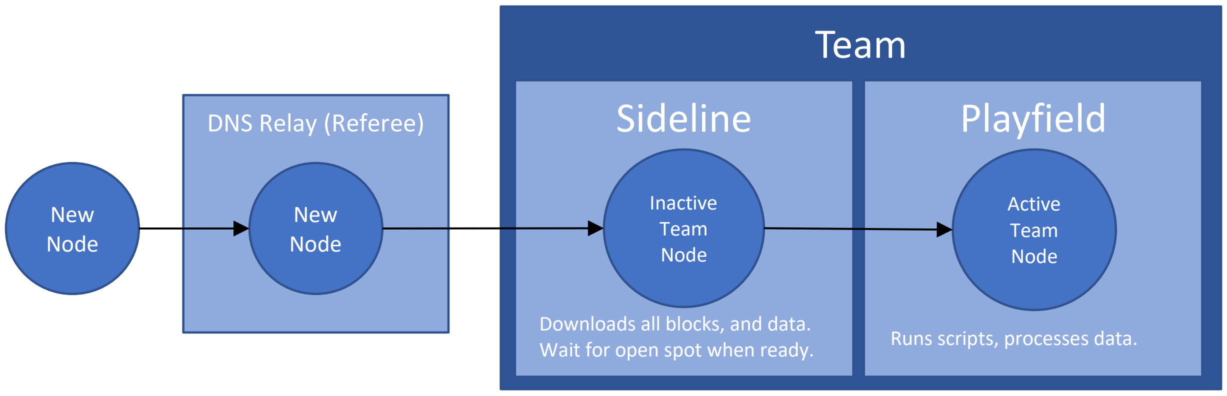
Since each team is completely scalable, the team sizes will vary. For every 25 nodes on a team, there will be 1 referee. Teams will be considered full when there are 100 nodes on a team, however this will increase as the team sees more traffic. Additionally, any new nodes joining the network are assigned to random teams by a referee based on the node's location. Ideally each team would have 1 node from every major location around the world. Currently, this is how the team structure looks:



The DNS relay directs traffic to the correct team's captain that has the smart contract or ledger the client may be looking for. The captain will redirect the traffic to another node executing the smart contract or ledger. If the DNS relay used is on the same team, it will redirect to another node.

Joining a team process

Any new nodes assigned to a team are placed into a queue (called the sideline), while on the sideline they download all blocks, verify them, and wait to be added to the playfield. The playfield is where all nodes reside who are earning coins and are add transactions to the blockchain. Nodes on the sidelines do not earn coins, but do help verify transactions. For every 10 scripts submitted to a team, the best performing node on the sidelines will be moved to the playfield where they will earn coins. Any nodes with network verified accounts attached on the sidelines that are being considered to move to the playfield will have preference to move.


Team
DNS Relay (Referee)
Sideline
Playfield
New Node
New Node
Inactive Team Node
Active Team Node
Downloads all blocks, and data. Wait for open spot when ready.
Runs scripts, processes data.

All referees keep track of the number of nodes on each of their respective teams along with the number of scripts executing. Once the maximum number of scripts are being executed on each team, all referees verify this with each other, then create a new team, where a temporary referee is placed. The referee selected is the one with the best ratings, or in the event of a tie, the one that has been a referee the longest. A referee can exist on two teams simultaneously. Once a permanent referee is selected for that team, the temporary referee returns to their original team.

Accounts

Wallets would be in the form of an account. An account can be created for free on any team, however the individual trying to create the account must verify a series of transactions and blocks which can take anywhere from 1 to 2 minutes depending on their computer. Or the individual can pay to create the account immediately for 0.1 XRJ which would be split up amongst all nodes on the playfield (including referees and captains). Accounts will only exist on that team and not the entire network. These accounts could hold XRJ coins or any other token created on the network, and would be able to transfer to other accounts on different teams. Each account would have a public key associated with it, and the user of the account would have to keep the private key secret. Accounts would take the form of an email, with the username first, then the team that account is a part of, then the coin symbol. An example would be: nathan@team10.xrj. If sending a token created on the network, for example called Test Coin (TEST), this would look like: nathan@team10.test.

The format is:

[Account name]@team[number].[token symbol]

Additionally, not only would tokens or coins be able to be sent, but objects would be able to be sent as well. This would be great for transferring game objects from one person to another, or even real-world real estate. An example would look like: nathan@team10.object->house. This would allow the transfer of objects from one smart contract or dapp to another securely, rather than having to rely on a smart contract to remain secure.

The format for this would look like:

[Account name]@team[number].object->[object symbol]

A url encoded query string could also be appended to the end of the account to send the transaction to. So a transaction to an another account could look like:

nathan@team10.object->house?id=19&name=Cool+house

or

nathan@team10.xrj?message=Hi+there+guy

Transactions

Each account has a public key associated with it, whenever a transaction or action from the account holder needs to be done, the account holder must sign a message using their private key detailing what the user would like to do. This message is then sent to the team's referee associated with the account, and the captain. The captain places a note as to where this transaction will reside on the merkle tree, then relays this to the referee. The referee and captain relay the transaction's message to the rest of the team, where the rest of the team build the merkle tree in unison according to the captain's notes. While the transaction's message is being confirmed across the team, the account is locked, and cannot send anymore transactions until all nodes on the team verify the transaction's message. Once all nodes on the team's playfield have verified and processed the transaction's message, the account is unlocked. If a node on the playfield is taking too long to respond (i.e. > 300ms), nodes on the sidelines can verify this transaction instead, increasing their chances of joining the playfield, while decreasing the chance of remaining on the playfield for the node that’s not responding.

While processing a transaction from one account to another, the tokens, or objects to be transferred are simply subtracted from the account, while added to the receiving account. What is stored on the ledger is that tokens were subtracted from one account, then another account received tokens. We may include an option where tokens are immediately removed from the sending account, then sent in random chunks later. To do this, multiple hashes that represent the process that must occur, would be sent to the nodes that must process the transaction that would be instructed to process the transaction at later times. Doing this adds a slight layer of transactional anonymity. We are still in the process of determining how to make each account’s holdings completely anonymous.

Should a transaction take place between accounts on different teams, referees on each of the teams contact each other and share the list of IPs on their respective teams. These IPs are then split up into chunks and passed to each node on each team where they then connect to each other a verifying the transaction.

All nodes throughout the network keep track of the total number of coins that are currently on the network, and that are on each team. As coins are subtracted and added based on each transaction, the total coin for the network and each team is updated as well. Referees continually share the latest coin count changes amongst each other, and pass these changes to the nodes as well.

The Merkle Tree and Blocks

There would be one merkle tree that contains all the objects and their hashes that describe what has occurred within that block. There would be no limitations to the block size, and a new block would be created by the captain every hour.

Network Verified Individuals (NVI)

To bring an additional layer of security to the network, individuals can submit a request to the network to go through a background screening process. This is not required, and no personal information would appear on the network, even in an encrypted form. The request submitted would simply appear on the network where the individual would have to contact one of the network selected background screening services and setup an appointment. Once the individual has gone through the process, the background screening service then updates that request with information based on what they found. Information would include:

- The country id the individual lives in
- Number of convicted crimes
- Number of generated NVIs

Once the request has been updated by the background screening service, that request now becomes a Network Verified Individual (NVI). This can be attached to a node. In the event the NVI becomes compromised, in other words, the NVI has had their information publicly available, a new NVI can be generated by a referee, and the old one destroyed. All data from the old NVI would be transferred to the new NVI. The number of generated NVIs would increase by 1 for each time this happens. After 5 generated NVIs, it is up to the referee to determine whether to generate a new NVI or not. The individual requesting a new NVI to be generated would have to pay a fee of 0.1 XRJ, which would go to the referee generating the new NVI. Only network verified individuals are capable of casting votes on the network. Additionally, we are considering requiring all nodes that wish to participate on the network to be ran by NVIs, however we are open to debate on this.

Captain Nodes

The goal of every node on a team is to become a team captain. The captain generates the main block hash based on all validated data from the merkle tree, then passes this block hash on to the rest of the team and referees for them to verify. Additionally, as transactions come into the team, the captain assigns some notes to the transaction, telling the rest of the team where to place the transaction object on the merkle tree.

Referees

To help manage the network, referees are needed to help ensure the network is playing fairly, help facilitate the creation of new teams, and route

Everything in this document is subject to change and is open for community participation

network traffic. Referees are partially human controlled, and will be selected by the network by voting for each referee. Any network verified individuals can apply to become a referee. Additionally, only referees can become DNS relays. Referee nodes must be capable of handling at least 30 open connections.

Locking Process

Referees will be able to lock accounts, smart contracts, dapps, and server apps only if a network selected cyber security organization has submitted evidence to a report, and this evidence has been verified by another cyber security organization. For more information about the rules that would cause an account to be locked, please see the section, Network Rules. If the account is locked, 10 referees are needed to review the evidence associated with the locked account to make a verdict. After the account is locked, referees have 2 days to make their decision. If they need more time or if more evidence needs to be collected, they can extend the decision period by a maximum of 14 days. Should 60% of the referees deem the evidence to be insufficient the account will become unlocked immediately after the decision. If a decision is not made within the decision period, or if the referees find the locked account to be not guilty, the account will become unlocked immediately. Should the referees find the locked account to be guilty, the account will go into appeal mode for the next 5 days. If within that 5-day interval, if at least 50 Network Verified Individuals (NVIs) find the referees decision to be in error, the verdict will be overthrown, and the account will become unlocked. If after the 5-day appeal expires, if the account has any coins inside it, 3 referees are needed to vote on where to move the coins. Options would include, return the coins to where they last came from, send the coins to a network selected charity, or move the coins to a specific account. If the last option is selected, then 3 more referees must vote on the specific account to return the coins to.

Checks and Balances

In a decentralized and distributed network, it would be foolish to trust anyone on the network, no matter their role. Hence why there must be systems in place to ensure everyone is playing correctly. Referees can be voted out should they receive too many bad reviews from other Network Verified Individuals (NVIs). Referee ratings would include:

- Friendliness (1 – 10)
- Properly routing traffic (1 – 10)
    - Nodes throughout the network must occasionally test each referee to ensure its routing properly
- Number of unjust account locks
    - If 50 NVIs have overthrown a verdict, this will increase by 1.

In the event a group of NVIs team up to overthrow verdicts consistently, especially in the face of overwhelming evidence, those NVIs will be permanently banned from participating on the network. This would require at least 5 referees and 10 other NVIs to vote on the list of individuals to ban.

Legal Representation

Additionally, should there be countries or jurisdictions where they do not allow the RadJav network to be legally represented, those countries will not be able to host nodes. Any NVIs within those countries will not be able to host a node, cast any votes on the network, become a referee, etc. Of course, they would be free to make transactions, create smart contracts, etc.

Voting

Typically, any vote that occurs must be conducted within 24 hours, unless otherwise noted in this document. Votes can only be casted by network verified individuals.

Network Rules

Freedom does not require total anarchy. With this in mind, we want to have a minimal set of rules set by the network that everyone on the network must abide by. People are free to produce what they want on the network so long as it does not violate the rules of the network. All rules are selected by the network and enforced by the network, no other government or organization has control over what these rules are, or how they are enforced. We the developers, will work with the community on what these initial rules are, however once the network is launched, we, the developers have the same voice as everyone else on the network. Our sole role is to provide code updates to the network, and RadJav as a software development platform. The starting rules for all smart contracts, dapps, and server apps are:

- Do not knowingly participate in the proliferation of sexual content involving those under the age of 18.
- Do not knowingly participate in the slave trade of human beings.
- Do not knowingly participate in murder for hire services.
- Do not knowingly participate in theft of any kind.
- Do not participate in attacking the RadJav network, any other device, or network. With an exception being that the target has requested to be attacked.

If a smart contract, dapp, or server app being hosted by the RadJav network is found to be breaking any of these rules, evidence will have to be collected by network selected cyber security organizations then submitted to a report where it's to be reviewed by a referee and an additional cyber security organization. After this has occurred, the Locking Process begins, for more information about this process please see the Locking Process section. Please note that the words "knowingly participate" are being used; this implies that the smart contract/dapp is knowingly involved in breaking that particular rule, and needs to be shut down. Higher Edge Software asks that whatever initial network rules that are selected at the time of the network launch, that they cannot be removed, only new network rules can be added, so long as they do not interfere with the initially selected rules.

Network Upgrades

The network would be coded using RadJav, with all code the same across all nodes. The initial codebase will be built into RadJav, but will be updated by the network as it comes to consensus on new code to use. All network upgrade proposals and code will be placed into a blockchain that's kept by all nodes. A new network upgrade proposal can be submitted by any NVI on the network. As a new network upgrade proposal is introduced, only nodes with a NVI attached that's on the playfield agree can cast a vote. With the new proposal these nodes can "move" into the proposal. This would essentially be a vote, and would lock that node into the proposal. Should 70% of the nodes on the network "move" into the proposal, Higher Edge Software will begin coding the proposal. Once the code has been written, we would submit it to the proposal for review. The code can then be reviewed and rated by NVIs in the following categories (there will be more):

- How well does it follow the proposal (0 – 10)
- How secure it seems (0 – 10)

All nodes that have "moved" into the proposal would then have the option to "move" out of the proposal if they don't like the code they see. Should 70% of the network remain in the proposal after 5 days of the code being introduced, then the rest of the network would be forced to upgrade. Should 31% of the nodes in the proposal need more time to review the code, they can delay the launch of the new code by 1 day for every day they vote. Higher Edge Software can delay the launch of the new code as well, for as long as necessary. We will only act as the code maintainer, and will have the same voice as everyone else on the network. Higher Edge Software only requires that whatever the mining payout for the developers that is agreed upon at the time of the network's launch does not decrease. Additionally, we require that whatever the initial network rules that are selected at the time of the network launch, that they cannot be removed, only new network rules can be added, so long as they do not interfere with the initial rules.

Mining Payouts

For every team on the network approximately 1.65 XRJ will be generated per hour. Assuming there's 200 teams with 100 nodes, and 4 referees on each team, approximately 2,890,800 XRJ would be generated per year. The math looks like this: 200 teams * 24 hours a day * 1.65 XRJ per hour * 365 days a year.

- 0.1 XRJ goes to the Captain
- 0.5 XRJ is split up amongst each node on the team (excluding the captain and referees)
    - Node payout = (0.5 * (Node performance multiplier)) / (Number of nodes on the team)

- 0.1 XRJ will go to the developers
- 0.1 XRJ will go to each DNS relay on the team
  - If there are no DNS relays on that team, this will be lost
- 0.05 XRJ will go to a pool for DDOS protection services
- 0.1 XRJ will go to a pool for bug bounties and security vulnerabilities
- 0.1 XRJ will go to a pool for the network's legal representatives to use when necessary
- 0.1 XRJ will be split up evenly amongst the background screening providers
  - Max 10 organizations
- 0.1 XRJ will be split up evenly amongst the cyber security organizations
  - Max 6 organizations
- 0.1 XRJ will go to each referee on the team to help maintain the network
  - Typically, there will be 4 referees per team
  - On some teams there will be more, some teams there will be less. It depends on the team size.

Node Performance Multiplier

The node's performance multiplier looks like:

- 5 points – Node has a NVI attached.
- 1 point – For every 10ms below 50ms in latency.
  - See the section Determining the Node's Latency for more information.
- 1 point – For every 2 days without disconnecting from the network for more than 5 minutes.
  - Maximum of 10 days.
- 1 point – For every node beaten to report the results from executing all scripts.
  - See the section Smart Contract/Dapp Execution for more information.

    - For nodes that have tied, no points would be awarded for the tied nodes.
    - Maximum of 10 nodes.
- 1 point – For every 100 smart contracts and dapps that are currently executing.
    - See the section Smart Contract/Dapp Execution for more information.
    - Maximum of 5,000 contracts.

Determining the Node's Latency

We are still determining the best way to determine a node's latency. Currently we are exploring the following options:

1. Have referees inform nodes as to who their closest neighbors are and have them ping each other, then report back to the referees what their found average pings times are. In this case, referees would have to assign geographic locations to each other to monitor.
    a. Pros: Get accurate latency readings for that given area.
    b. Cons: If a single user owns a large number of nodes within a single area, they could report the best latency at all times. We could implement a counter-measure where nodes operated by the same NVI would be banned from pinging each other.
2. All nodes within a team ping each other reporting back to their referees what their found average times are.
    a. Pros: Easy to implement.
    b. Cons: Nodes on a team will be on different parts of the Earth and nodes located physically farther away from the rest of the team will always be at a disadvantage.
3. All nodes on a team can have "sub-nodes" where essentially a single node can contain multiple sub-nodes, but are located in different locations around the Earth. Referees would note the geographic locations of these nodes in relation to other nodes on the same team and have the closest nodes ping each other, then report back.
    a. Pros: None.

b. Cons: Could be difficult to implement. Additionally, it could take a lot away from the network's scalability, since a single node could be multiple nodes.

Any suggestions or comments are welcome. Currently we are thinking about implementing options 1 or 2.

Smart Contract/Dapp Execution

For now, let us define a script as smart contract or dapp. Three threads will be created, with a maximum of 2,500 scripts in each. Unless the script has paid extra to run on its own thread, every script will be placed into a single thread where an update function will be executed. All code executed within that function must be executed within 400 microseconds, else it will be paused until the next update. Should a script need more processing power or transactions to be processed, the developer of that script would be able to have it switch to execute on its own thread at any time for an additional fee.

Memory each script is using is saved to disk as its used. Should a script need faster access to RAM, the script can reserve RAM for use.

During a node's start up process, the node would have to test how many threads it can handle executing at once, then set its own maximum thread count. Once that maximum thread count is reached, no additional threads can be created, and this is reported back to the referees. Once the maximum thread count for all executing scripts has been reached across the entire team on the playfield, the team no longer accepts new scripts, and the referees must determine where to route any newly submitted scripts.

Every line of code that's executed is added up and hashed to prove that code was executed. For every 1000 lines of code executed, the totaled hash is packaged up with the other hashes from all the other scripts and is sent to the referees first for them to determine who executed it faster.

Every 2 minutes all nodes on the team are synced up again by the referees. This allows any new nodes coming along to start competing, and any nodes that were temporarily knocked offline to catch up again. This works by pausing scripts one at a time and have the node that most recently reported to the referees, report its current line position. Then all other nodes must pause once they reach that line number. Once all nodes scripts are paused, then execution starts again 200ms after the referees send the go signal. This occurs on a script-by-script basis.

All transactions or messages being sent to the scripts are pushed and popped on a stack as necessary.

Smart Contract/Dapp Fees

Smart contracts and dapps would be able to reserve the maximum amount of RAM, threads, and storage to use.

Public smart contract/dapp fees would look like:

- Base fee: 0.000001 XRJ
    - The base fee is raised by 0.0000001 for every 10 new teams created.
- Usage per hour: 2* base fee
- Data transfer in: 4 * base fee
- Data transfer out: 5 * base fee
- Data storage: 3 * base fee, per 100MB stored
- Reserve thread: 1000 * base fee, per thread
- Reserve RAM: 100 * base fee, per 1 MB

Private smart contract/dapp fees would be the same except:

- Base fee: 0.000001 XRJ
    - The base fee is raised by 0.0000001 for every 10 new teams created.

Scripting Language

A simple-typed scripting language will be created that would allow for deterministic code to be executed. This scripting language for now is called “Slide”. This language would be used to create the smart contracts and dapps, additionally it possibly may be used as an alternative to TypeScript to transpile JavaScript.

Smart Contract and Dapp Differences

We believe smart contracts should be like those of real world contracts. Typically, real world agreements are between parties and are fairly straight forward with a list of rules each party is agreeing to. There's typically no recursion, no abstract classes, few if any calls to any other documents, etc. In contrast, smart contracts today are very complex and can wind up inadvertently creating security vulnerabilities. The worst possible scenario is to create a smart contract that has millions of dollars stored in it, to have a security vulnerability. Especially when states, such as our home state Arizona is beginning to make smart contracts legally binding. Smart contracts need to be clear and understandable.

Therefore, any smart contracts created with Slide would be very restrictive. There would be no recursion, very limited OOP, no nested loops, very restricted (if any) calls to other smart contracts, and the call stack would remain small. Even without all these features, it would still be Turing Complete. Code reviewing a RadJav smart contract should be a relatively easy and quick task.

Dapps on the other hand would have no limitations of any kind. Any type of dapp would be able to be created, then it is up to the users to determine whether or not they trust the dapp they're putting their coins into.

Tokens/Objects

The tokens created on the network would remain only on that team, however the symbol associated with the token would be stored in a database amongst all the referees, so they can route traffic as needed. The same for any objects created, a symbol would be registered for that object which would be stored amongst all referees to direct traffic.

Token symbols and object symbols can be reserved for a fee of 0.01 XRJ per month or 0.11 XRJ per year. Additionally, for tokens to be used on the network it would cost 0.00001 XRJ per hour for up to 40,000 transactions per hour. For every 40,000 transactions per hour, the fee has a multiplier that is increased by 1. For example:

- Hour 1: 30,000 tx/h - 0.00001 XRJ/h
- Hour 2: 40,000 tx/h - 0.00002 XRJ/h
- Hour 3: 40,000 tx/h - 0.00002 XRJ/h

- Hour 4: 70,000 tx/h - 0.00002 XRJ/h
- Hour 5: 80,000 tx/h - 0.00003 XRJ/h
- Hour 6: 90,000 tx/h - 0.00003 XRJ/h
- Hour 7: 120,000 tx/h - 0.00004 XRJ/h

Debugging Smart Contracts/Dapps

Smart contracts and dapps submitted to the network can be submitted as a debug or release script. If the script is submitted as a debug copy, it will have a public key associated with it that allow developers to debug their application in real time as it executes communicating with the developer’s machine. However, if its released as a release copy, then it will not be able to be debugged.

Smart Contract/Dapp Security

Users that will be interacting with smart contracts or dapps should look at the script through an online block explorer and ensure that they will be interacting with a smart contract that is in release mode. This should help ensure users that the smart contract they will be working with is in fact secure.

In a future release, we would like to have developers submit their organizations to the network with their public key signatures, so uploaded smart contracts can be signed by these organizations. Thus, helping everyone identify that is in fact the correct smart contract they will be interacting with.

Network Selected Organizations

The following organizations can be submitted to the network:

- Background Screening Providers
- Cyber Security Organizations
- DDOS Protection Services
- Law Firms
- Charities

Any of the listed organization types can apply to be listed on the network. They must verify a series of transactions first that lasts up to 1-2 minutes to be listed for free, or pay a fee of 0.1 XRJ that would be split up amongst all referees.

© FogChain, Corp

Once an organization has applied to be listed, only NVIs can rate the trustworthiness of the organization in multiple categories. After the organization has received over 50 positive ratings, then the organization will be listed on the network, and can be voted on. The categories and their values can be determined with the community. However, for now, the categories to be voted on would be:

- Is it a trustworthy organization (On a scale from 1-10)?
- Number of NVIs that has verified that this is the organization being listed.

Network Selected Background Screening Providers

The background screening providers would verify that the individual applying to the network to become an NVI is who they say they are. Not to mention this provider would do a quick background check to make sure they are not a convicted criminal. The idea is they would do a video call with the individual who is applying and have them do a live public key signing while on video to ensure that is who the NVI claims to be. Additionally they can verify the country of residence and that the individual resides in. If the country the NVI resides in does not allow the RadJav network to be legally represented there, unfortunately that NVI would not be able to participate as a NVI on the network.

Once the background screening provider has verified the identity of the individual they can sign their NVI stating that they have been identified. The basic personal information of the individual, such as first name, last name, and current address would be saved and encrypted to be stored on the background screen providers computers. If the community does not like this idea, then that data can be transferred to be stored on a network selected high security data storage facility that would be paid out of another smaller pool.

There could only be a maximum of 10 selected background screening providers.

Network Selected Law Firms

Selected law firms would be able to be hired by the network to represent the network anywhere in the world. For example, if the network were to come under attack from a group of hackers, the network selected cyber security organization can attempt to find them. If they are caught, the network could hire a legal team within that country to press charges against them. Or, if some unjust regulations were trying to be passed somewhere in the world, lawyers or lobbyists within those countries could be hired to help prevent those regulations from passing. Even legal packages for ICOs could be drafted for each country people wish to host their ICO in.

Submitting legal requests

Any NVI on the network can submit a legal request, with a selected law firm, to the network for other NVIs to vote on. Should at least 50 NVIs vote in favor of the legal request, the selected law firm would then have an estimate requested. Once the law firm responds to the legal request with their estimate, at least 50 NVIs would have to vote in favor of paying the estimate, along with any monthly fees listed with the estimate. Should the 50 NVIs vote in favor of the estimate, the correct amount of XRJ would be sent to the law firm for them to start.

Network Selected Cyber Security Organizations

The network selected cyber security organizations can help prevent hackers from attacking the network, or present evidence to referees to stop malicious activities from taking place on the network.

There could only be a maximum of 6 cyber security organizations selected.

Cyber Attack Events

In the event the network or a team comes under attack, a cyber security organization can issue a warning to the network. In the extremely unlikely situation that an attack is successful, the cyber security organization can present the evidence to the network for referees to review. Once 10 referees review the evidence, those referees can shut down a team temporarily from operating and the cyber security organization can work with us, the developers, to attempt to fix the problem. If the attack has affected the entire network, 80% of all referees on the network would have to vote to shut down the network temporarily to fix the issue. A successful attack on a network this large and secure is extremely unlikely, but when millions or billions of dollars are at stake, every precaution is necessary.

Network Selected DDOS Protection Services

The network can select a DDOS protection service to help protect all DNS relays, and nodes on the network.

Network Selected Charities

Selected charities would be able to be listed on the network for people to donate to whenever they wish. Or in the event some XRJ is confiscated, that confiscated XRJ can be instead donated to a random charity.

Bug Bounties/Security Vulnerabilities Pool

To ensure the security of the network and RadJav as a software development platform, a constant bug bounty will be in place at all times. Any bug found can be submitted by anyone on the network, then NVIs can determine how valuable in XRJ that bug is, and how valuable the solution for that bug is. We the developers will look at the submitted bugs, and implement their fixes. Once we, the developers, look at the bug submission and determine if its valid, we'll mark it as valid, and the network can then pay the correct amount of XRJ the network has agreed to pay. Then we the developers implement that fix. Should the price of the bug submission be too high, 2 referees would be needed to clear the XRJ pay out, then NVIs would have to vote again as to how much should be paid.

V2 Frontend for Desktop and Mobile Devices

Creating the UI for a platform of this size would be a nightmare with anything else but RadJav. The reason why this network can be built in a timely manner is because it would be built using RadJav itself. Since RadJav is easy to develop with, and will only get easier to develop with, creating a network of this size is easier. Desktop and mobile apps would be created that would allow NVIs to interact with the RadJav V2 network quickly and easily. Notifications could appear on the users phone asking the user to vote on a particular topic.

© FogChain, Corp

Everything in this document is subject to change
and is open for community participation