Red: a full-stack, open-source toolchain for simple smart contracts and decentralized apps development

Nenad Rakocevic, *nenad@rakocevic.net*, Tamás Herman, *tamas@herman.im*

December 2017

WORKING DRAFT

ABSTRACT

Blockchain technologies will dramatically alter the landscape of many industries and the way human activities are managed, using a fair and trustworthy decentralized model. Smart contracts [1] are at the center of this revolution, yet contract development tools are crude, hindering the adoption and reliability of this new technology. The innovative Red programming stack[2] offers a better solution , which will greatly simplify smart contract development and improve their security. A new token, named Red Community Token (RED), has been created to fuel the growth of the Red open source programming stack and its community. The Red toolchain will be extended to ease the development of more secure smart contracts and Decentralized Applications [3] (Dapps) superseding current, inadequate tools and their complex architectures.

1. MISSION STATEMENT

The mission of the Red team is to power the revolution promised by blockchains, smart contracts and decentralized apps, by lowering the barrier for smart contracts and Dapps creation and deployment using the innovative Red fullstack solutions.

2. STATE OF SMART CONTRACT DEVELOPMENT

Smart contracts, as general-purpose computations taking place on a blockchain, are the most promising part of blockchain technologies. They provide a programmable "partially or fully self-executing, self-enforcing" model, with different levels of capabilities [4] depending on the blockchain type (Bitcoin, Ethereum, NEO,…). Such contracts are currently implemented using first-generation programming languages, capable of compiling to the bytecode used by the chains VM (Virtual Machines). The most prominent smart contracts platform is Ethereum. Solidity [5] is the language used most to program smart contracts on Ethereum.

Solidity is programming language for smart contracts with general-purpose capabilities. It allows the implementation of any kind of computation on the blockchain, at the price of high complexity and high security risks. Any mistake in a Solidity contract can lead to partial or total loss of Ether controlled by that contract when the security flaws are exploited by attackers. The only counter-measure is submitting the contract's code to Solidity experts for auditing before deploying. This is both a slow and expensive procedure, which results only in reducing the risks, not suppressing them. Moreover, Solidity has notorious design issues [6] [7] [8] that are further undermining smart contract development.

Another possible approach is to use templates for contracts, and parameterize them for a given use-case. This approach theoretically mostly eliminates the security issues, as the contracts code is always the same, so it can be audited extensively before been put into production. The drawback is that the application domain then becomes extremely limited, so that template providers need to produce a vast quantity of templates, covering the most used cases, in order to stay relevant. Still, with the advertised "no-programming" contracts, the "general-purpose computation" aspect of the blockchain is lost in the process, as well as the decentralization (template platforms are mostly controlled by single entities[9]) at the core of the blockchain philosophy. Templates could help smart contracts gain more popularity and

traction in the early days, but they are not the usual way software is built, so this approach might never become mainstream.

When the smart contracts are combined with a User Interface, they are then called "Decentralized Apps" (Dapps). Dapps are currently built as web apps, served online through a browser, or encapsulated as a standalone, downloadable executable (typically using Electron framework, like Ethereum's official client, Mist). By running on a web stack, Dapps suffer from several issues:

- Security issues: the web stack, being so pervasive nowadays and having an extremely vast attack surface (25+ millions of lines of complex code), is a big target for attackers. When Dapps are distributed as standalone clients, constant updates are required to fix the security flaws and bugs of the underlying web stack[10] [11].
- Heavy deployment: Dapp user interface are running in a webview (browser or encapsulated in Electron), but still require separate download and installation of a blockchain light client or node[12], making it a complex and/or expensive barrier for non-experts users.
- Webapp development, despite being in broad usage, still has a long learning curve as one need to master at least three different languages (HTML/CSS/JS), and a vast amount of quickly changing frameworks, feeding a growing complexity[13].

All those issues described above, are real obstacles to the faster adoption of blockchain technologies. The Red team intends to tackle them with the innovative Red language stack and toolchain.

3. NECESSITY OF A LANGUAGE FOR BLOCKCHAINS

Blockchains are networks of virtual machines running specialized code. The ability to simply and efficiently create smart contracts and decentralized apps is critical to the success of any blockchain beyond just the speculative usage of its intrinsic fuel token. Templating solutions are barely scratching the surface of possible contracts use-cases, only a programming language can fulfill the promise of the "decentralized world computer" made by advanced blockchains like Ethereum. Red will be the powerful enabler that allows a new wave of smart contract programmers and users to emerge. It will create opportunities for people that are currently only available to very well resourced individuals and organisations. Just as tools like Visual Basic for desktop development in the hands of many people, Red will do the same for smart contracts thanks to its simplified and effective programming paradigm.

4. THE RED FULLSTACK APPROACH

Red is an open-source programming language launched in 2011 by Nenad Rakocevic. Its goal is to provide a simpler and more cost-effective way to build software. Red is a very high-level language, capable of extremely expressive code[14].

Besides the features [15] that Red language can already accomplish, the biggest leverage provided by Red is the ability to easily implement Domain-Specific Languages (DSL). DSL are one of the most effective ways to reduce complexity in software development and Red uses them in a pervasive way. Several ones are already included to cover GUI programming, 2D drawing, pattern-matching, data extraction and system programming.

Thanks to those DSL, Red covers the whole range of software abstraction layers, and extends now horizontally across domains.

The Red toolchain is a zero-install, zero-configuration, single 1MB file [16], containing its complete toolchain including a cross-platform native compiler, the whole standard library (more than 50 datatypes), 5 DSL, a cross-platform native GUI system and an interactive console. The codebase for all those features weighs about 100k LOC.

As an open-source project, with a transparent mode of operation, Red has a growing community of thousands of users and dozens of contributors working daily to improve Red and help newcomers learn it. The Red project already has 2960 stars on Github, 1.6M views on its web site and about 530

developers in the online chat room[17] discussing about Red daily (at end of Dec, 2017).

5. RED FOR SMART CONTRACTS

Given the current obstacles that smart contracts and Dapps are facing, the Red stack is a perfect fit to help pass them and accelerate world-wide adoption of software powered by blockchains.

In order to realise that, the Red team will be providing a uniquely easy to use, integrated set of tools for smart contract development. This will include a smart contract programming language, a complete toolchain for building smart contracts and a library for building Dapps.

6. RED/C³, A LANGUAGE FOR SMART CONTRACT PROGRAMMING

Red Cross Chain Code (Red/C³) is the Red DSL for cross-chain programming and the key part of our solution. It will compile directly to the blockchain Virtual Machine (VM), using different backends targeting each blockchain (starting with Ethereum). The language itself is composed of two layers:

- A very high-level layer: a symbolic, highly declarative language with restricted computing semantics and restricted control flow (turing-incomplete), reading mostly like natural language (compared to mainstream languages). Coarse-grained code produced at this level by the compiler will exhibit higher safety, close to the level provided by templates, while still providing much greater flexibility. An intelligent, visual program builder will also be considered for pre-generating such code.
- A lower-level layer: a general-purpose, statically typed, subset of Red language, capable of general computations.

Contracts will be written in high-level code with the ability to locally drop to the lower layer for more general computations. Both layers will be able to call external contracts written using any other tools, so they will be fully interoperable with existing contracts. Using the new Simplicity [18] low-level language as intermediary representation is considered. Red/C³ is currently still under design.

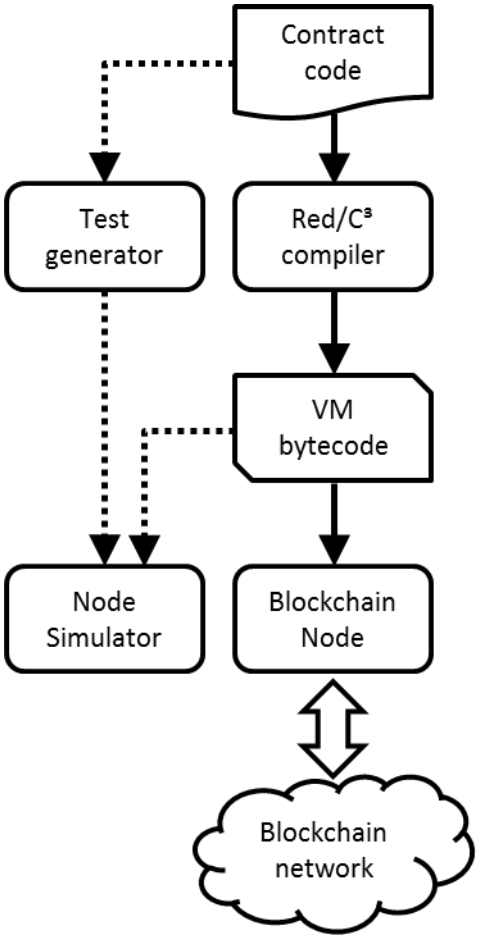
7. RED TOOLCHAIN FOR BLOCKCHAINS

The Red/C³ toolchain is a standalone toolchain, extending the existing Red toolchain, that can be run natively on users' desktop or mobile devices. It will include:

- A compiler to VM bytecode, with one backend per VM (starting with Ethereum VM).
- A generative test module for extensive contracts testing.
- Blockchain VM simulators written in Red/System[19] for fast performance (starting with Ethereum VM target).
- Wrappers to separately downloadable minimal blockchain nodes for real deployment.

The Red/C³ compiler will be written in Red language, with no dependency on any other toolchain. The compiler's implementation main concerns are simplicity and shortness (a few thousand LOC is the target) in order to ease auditing and future formal proving of correctness.

Fig.1: Red/C³ toolchain overview



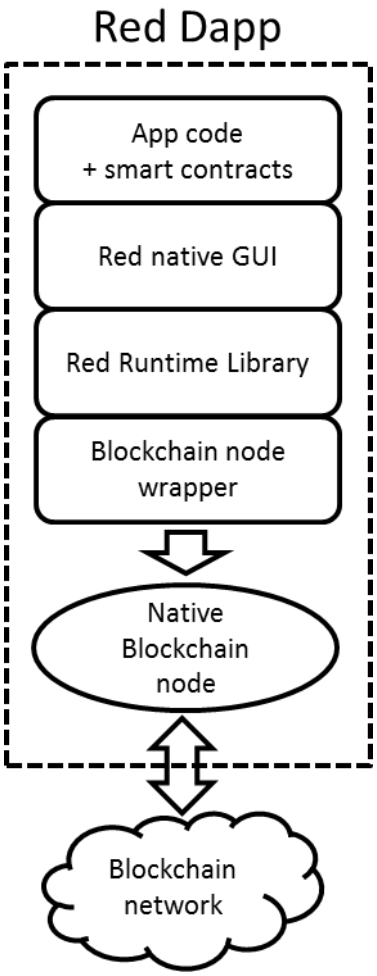
8. RED DAPPS

The web stack used so far in Dapps will be replaced by the Red one, which will be about 100 times smaller for standalone Dapps (1MB for Red's runtime vs ~120MB for Electron) and will provide an *integrated* native blockchain node, avoiding complex deployment requirements, that plague current Dapps. The Red regular language and all its built-in DSL will be fully available for Dapps building. The Red standard library will be extended to include:

- a wrapper for integrating a targeted blockchain node (using direct C API and/or JSON-RPC for interfacing).
- a high-level interface to operate the smart contracts and send transactions at run-time from regular Red code (equivalent to web3 or similar interfaces in web Dapps[20])
- a sandboxed I/O mode to enforce security on devices running Dapps.
- a secure way to store and handle private keys required for sending transactions on the chains, including hardware wallet integration.
- a specific library for RED token management (for easing and encouraging RED-oriented Dapps creation), possibly based on AragonOS or zeppelin_os solutions[21].

The hosting and distribution model of such Red-powered Dapps is not yet decided. Leveraging the decentralized Swarm[22] and IPFS[23] infrastructures for those is strongly considered.

Fig.2 Red Dapps runtime stack



9. RED: RED COMMUNITY TOKEN

The community has been a very important component of the whole Red project since the beginning. We are now willing to go one step further, and empower the Red community even more by

making it a Distributed Autonomous Organisation (DAO). This DAO will be enabled by a community-oriented transferable ERC-20 token called Red Community Token (RED). Holding RED tokens will be the pre-requirement to be a member of the Red DAO. The membership will provide privileges, including but not limited to, voting rights over different aspects of the whole project, such as prioritizing new features and bug fixes. Transactions between community members themselves, including but not limited to, tipping other developers contributions and intra-community services will require RED tokens. The Red community will pave the way of a new economical model for open source projects communities, building community tools powered by Dapps and RED tokens, which could be used later on, by other open source communities.

In order to achieve these goals, all the Red open source project's assets will be moved to a non-profit foundation, which will define the rules of RED usage. We are closely following how the OAX foundation[24], Aragon and other DAOs are operating and will seek partnerships to help build an optimal organization. The voting rules and intra-community modes of operations for the RED will be defined by the foundation.

Specific Dapps using Red stack will be built, allowing deeper interconnection between community members. Such "community Dapps" would require RED tokens to operate. They would both power the community economics and serve as showcase apps of what the Red programming stack can accomplish. They would include, but not limited to:

- a RED wallet Dapp
- a community chat system
- a community code exchange place
- collaborative code editor

Providing more community Dapps will be actively encouraged by the foundation through a rewarding program using RED tokens:

Foundation to community:

- rewarding code contributions
- rewarding community management
- rewarding Red learning tutorials completion
- rewarding marketing actions (events organization, presentations and promotions)

Community to community:

- tipping chat posts
- tipping code contributions
- tipping direct peer to peer coding help and code snippet exchanges
- tipping community-provided learning materials
- paying for peer to peer services, like app development, code reviewing and debugging, assets (graphic/video/audio) design and editing, tutoring.

In the future, the RED token is also intended to serve as a "meta-fuel" token for smart contract execution across different blockchains, to abstract the different per-chain intrinsic fuel tokens.

10. THE INITIAL COIN OFFERING

In order to kickstart the Red community economics and the new blockchain-related technical developments for Red runtime stack and toolchain, an ICO will be held to cover the costs of hiring extra talents, and marketing campaigns, services and products.

Detailed information about the ICO will be announced on the Red website: https://ico.red-lang.org.

11. ROADMAP

The development schedule prioritizes for Dapps building as soon as possible, in order to kickstart the community infrastructure and usage of the RED token. As the Red stack is already in a usable state on both Windows and macOS platforms, the only missing part is the blockchain node wrapper in the Red runtime library, so this is the first task to complete and starting with the Ethereum network.

Q1 2018:

- Ethereum node wrapper for Red Dapps (alpha)
- RED wallet Dapp (alpha)

Q2 2018:

- Red/C ³compiler (alpha 1)
- Node simulator (alpha)
- Bitcoin node wrapper for Red Dapps (beta)
- REDwallet Dapp

Q3 2018:

- Red/C ³compiler (alpha 2)
- Node simulator (beta)
- Community chat Dapp

Q4 2018:

- Red/C ³compiler (alpha 3)
- NEO node wrapper for Red Dapps (alpha)
- Community code exchange place

Q1 2019:

- Red/C ³compiler (beta)
- Collaborative code editor

Q2 2019:

- Red toolchain for Blockchains 1.0

12.LEGAL CONSIDERATIONS

The RED tokens usage will be deeply rooted in the intra-community operations and economics. RED tokens are not for speculative investment. No promises of any particular value or future performance of the RED are made. The Red stack provides integration with blockchain nodes, written by third-parties (like the Ethereum foundation), so risks associated with blockchain usages and especially smart contract deployment, buying or selling smart contracts using Red platform or provided Dapps are assumed solely by the user.

THIS DOCUMENT AND ANY OTHER DOCUMENTS PUBLISHED IN ASSOCIATION WITH THIS WHITE PAPER RELATE TO A POTENTIAL TOKEN OFFERING TO PERSONS (CONTRIBUTORS) IN RESPECT OF THE INTENDED DEVELOPMENT AND USE OF THE NETWORK BY VARIOUS PARTICIPANTS. THIS DOCUMENT DOES NOT CONSTITUTE AN OFFER OF SECURITIES OR A PROMOTION, INVITATION OR SOLICITATION FOR INVESTMENT PURPOSES. THE TERMS OF THE CONTRIBUTION ARE NOT INTENDED TO BE A FINANCIAL SERVICES OFFERING DOCUMENT OR A PROSPECTUS. THE TOKEN OFFERING INVOLVES AND RELATES TO THE DEVELOPMENT AND USE OF EXPERIMENTAL SOFTWARE AND TECHNOLOGIES THAT MAY NOT COME TO FRUITION OR ACHIEVE THE OBJECTIVES SPECIFIED IN THIS WHITE PAPER. THE PURCHASE OF TOKENS REPRESENTS A HIGH RISK TO ANY CONTRIBUTORS. TOKENS DO NOT REPRESENT EQUITY, SHARES, UNITS, ROYALTIES OR RIGHTS TO CAPITAL, PROFIT OR INCOME IN THE NETWORK OR SOFTWARE OR IN THE ENTITY THAT ISSUES TOKENS OR ANY OTHER COMPANY OR INTELLECTUAL PROPERTY ASSOCIATED WITH THE NETWORK OR ANY OTHER PUBLIC OR PRIVATE ENTERPRISE, CORPORATION, FOUNDATION OR OTHER ENTITY IN ANY JURISDICTION. THE TOKEN IS NOT THEREFORE INTENDED TO REPRESENT A SECURITY INTEREST.

The most recent version of this document can be found at the following address:

https://ico.red-lang.org/RED-whitepaper.pdf

References

[1] Smart contract https://en.wikipedia.org/wiki/Smart_contract

[2] Red language http://red-lang.org

[3] Decentralized Apps http://www.ethereumwiki.com/ethereum-wiki/dapps/

[4] Depending on their turing-completeness levels.

[5] https://en.wikipedia.org/wiki/Solidity

[6] "Why Solidity isn't solid"
https://medium.com/@Hibryda/why-solidity-isnt-solid-3341af77fc1c

[7] "A security issue with Ethereum's Solidity language, not just the DAO"
https://medium.com/@muneeb/solar-storm-a-serious-security-exploit-with-ethereum-not-just-the-dao-a03d797d98fa

[8] Solidity design issues: https://news.ycombinator.com/item?id=14691212

[9] like EtherParty or BlockCAT.

[10] https://github.com/ethereum/mist/releases

[11] «Electron apps are much easier to mess with! »
https://medium.com/@homakov/why-you-shouldnt-worry-about-the-jaxx-hack-6344a4c4a11

[12] like Mist client (about 60MB download) or using MetaMask browser extension. Light clients are still in very early stage though.

[13] https://hashnode.com/post/is-modern-front-end-overcomplicated-cipwgcbot06g10w537pbuytw5

[14] Red is the next evolution of Rebol language, most expressive general-purpose language according to Redmonk study: http://redmonk.com/dberkholz/2013/03/25/programming-languages-ranked-by-expressiveness/

[15] Parse DSL : http://www.red-lang.org/2013/11/041-introducing-parse.html
Cross-platform native GUI system : http://www.red-lang.org/2016/03/060-red-gui-system.html
Reactive programming: http://www.red-lang.org/2016/06/061-reactive-programming.html
Embeddability: http://www.red-lang.org/2017/03/062-libred-and-macros.html

[16] http://www.red-lang.org/p/download.html

[17] Red main chat room: https://gitter.im/red/red

[18] Simplicity, a low-level language for blockchain contracts: https://blockstream.com/simplicity.pdf

[19] Red/System language: http://static.red-lang.org/red-system-specs-light.html

[20] web3 (https://github.com/ethereum/web3.js), ethjs (https://github.com/ethjs/ethjs) or ether pudding (https://github.com/ethers/ether-pudding)

[21] AragonOS (https://blog.aragon.one/introducing-aragonos-say-hi-to-modular-and-extendable-organizations-8555af1076f3), zeppelin_os (https://zeppelinos.org)

[22] Swarm: http://swarm-gateways.net/bzz:/theswarm.eth/

[23] IPFS: https://ipfs.io/

[24] Open Asset Exchange: https://oax.org