KADENA

SCALABLE BLOCKCHAIN | SMARTER CONTRACTS

Pact™ & Smart Contracts: A 1-Page Overview

Version 1.1 - 2018 October

What are smart contracts?
"Smart contract" generally means *an app that runs on a blockchain*, and they were originally pioneered by the Ethereum Project. Their key idea was that if a blockchain were considered trustworthy enough to track coin ownership, why not extend that system to allow people to perform other delicate tasks as well? With smart contracts, a blockchain becomes a platform where users can create their own apps and offer them to the world.

Why do smart contracts even matter?
The programming languages used to write smart contracts today **are remarkably unsafe and full of catastrophic bugs**. Recently (April 2018), a bug in Ethereum's smart contract language *Solidity* allowed people to freely create uncountably large amounts of tokens, causing crypto exchanges to temporarily halt their services. This is not the first time this has happened to Solidity:

- November 2017: A programmer accidentally deletes some code, freezing approximately $150 million of Ether held in a widely used Parity wallet;
- July 2017: Cyber criminals break into Parity wallets, getting away with approximately $30 million of Ether; and
- June 2016: The Distributed Anonymous Organization (DAO), another Ethereum-based initiative, is hacked through its smart contracts, and the perpetrators escape with $60 million of Ether.

What is Pact?
Pact is a programming language for writing smart contracts that are free from the issues present in other languages. It was designed by Stuart Popejoy for **Kadena**, a blockchain platform provider.

Stuart cofounded Kadena with his fellow JP Morgan colleague Will Martino, and built an enterprise-grade private blockchain in 2016. The company recently raised $12M for Kadena's new public blockchain, Chainweb. Both of these blockchain products process smart contracts written in Pact.

How does Pact solve the issues in other smart contract languages?
Pact was intentionally designed with safety and security in mind. Unlike Solidity, Pact's code is human-readable. It is installed onto the blockchain directly, so you can always review the running code and be sure of what is happening. Furthermore, Pact allows you to *upgrade* your smart contracts, so that you can offer new features and fix bugs as they are discovered. None of this is currently possible with Solidity.

Most importantly, Pact's newest feature, **Formal Verification** (released in June 2018) allows users to *mathematically prove that their code does what it claims.* And, it's already open-source.

What is Formal Verification?
In high-level terms, it's a method of compiling code into a mathematical model, that can then be used to prove that the code contains no bugs. Formal Verification systems are used in nuclear power plants, aerospace autopilot systems, and now, with Pact, in a blockchain.

Why is this important?
Businesses and governments that are currently exploring blockchain technology cannot put their faith in a platform like Ethereum, with the potential catastrophic consequences and lost currency. With Kadena's smart contract language, Pact, users can trust that their code is protected by the same mathematical systems that protect mission-critical ventures, developed by technologists with over a decade of experience building systems for financial institutions.

To speak with Kadena's team about Pact, Formal Verification, or the state of blockchain, contact info@kadena.io (ATTN: Vivienne Chen, Media Manager)
For specifics of Pact's features and implementation, see the full Pact Whitepaper.