Nick Szabo's Papers and Concise Tutorials

The Idea of Smart Contracts

What is the meaning and purpose of "security"? How does it relate the the relationships we have? I argue that the formalizations of our relationships -- especially contracts -- provide the blueprint for ideal security.

Many kinds of contractual clauses (such as collateral, bonding, delineation of property rights, etc.) can be embedded in the hardware and software we deal with, in such a way as to make breach of contract expensive (if desired, sometimes prohibitively so) for the breacher. A canonical real-life example, which we might consider to be the primitive ancestor of smart contracts, is the humble vending machine. Within a limited amount of potential loss (the amount in the till should be less than the cost of breaching the mechanism), the machine takes in coins, and via a simple mechanism, which makes a freshman computer science problem in design with finite automata, dispense change and product according to the displayed price. The vending machine is a contract with bearer: anybody with coins can participate in an exchange with the vendor. The lockbox and other security mechanisms protect the stored coins and contents from attackers, sufficiently to allow profitable deployment of vending machines in a wide variety of areas.

Smart contracts go beyond the vending machine in proposing to embed contracts in all sorts of property that is valuable and controlled by digital means. Smart contracts reference that property in a dynamic, often proactively enforced form, and provide much better observation and verification where proactive measures must fall short.

As another example, consider a hypothetical digital security system for automobiles. The smart contract design strategy suggests that we successively refine security protocols to more fully embed in a property the contractual terms which deal with it. These protocols would give control of the cryptographic keys for operating the property to the person who rightfully owns that property, based on the terms of the contract. In the most straightforward implementation, the car can be rendered inoperable unless the proper challenge-response protocol is completed with its rightful owner, preventing theft.

If the car is being used to secure credit, strong security implemented in this traditional way would create a headache for the creditor - the repo man would no longer be able to confiscate a deadbeat's car. To redress this problem, we can create a smart lien protocol: if the owner fails to make payments, the smart contract invokes the lien protocol, which returns control of the car keys to the bank. This protocol might be much cheaper and more effective than a repo man. A further reification would provably remove the lien when the loan has been paid off, as well as account for hardship and operational exceptions. For example, it would be rude to revoke operation of the car while it's doing 75 down the freeway.

In this process of successive refinement we've gone from a crude security system to a reified contract:

(1) A lock to selectively let in the owner and
exlude third parties;
(2) A back door to let in the creditor;
(3a) Creditor back door switched on only upon nonpayment
for a certain period of time; and
(3b) The final electronic payment permanently switches
off the back door.

Mature security systems will be undertaking different behavior for different contracts. To continue with our example, if the automobile contract were a lease, the final payment would switch off leasee access; for purchase on credit, it would switch off creditor access. A security system, by successive redesign, increasingly approaches the logic of the contract which governs the rights and obligations covering the object, information, or computation being secured. Qualitatively different contractual terms, as well as technological differences in the property, give rise to the need for different protocols.

(Derived from "Formalizing and Securing Relationships on Public Networks" , by Nick Szabo)

A related article discusses a formal language for analyzing contracts and specifying smart contracts.

Please send your comments to