Smart Contracts: Building Blocks for Digital Markets

Glossary

(This is a partial rewrite of the article which appeared in Extropy #16)

Introduction

The contract, a set of promises agreed to in a "meeting of the minds", is the traditional way to formalize a relationship. While contracts are primarily used in business relationships (the focus of this article), they can also involve personal relationships such as marraiges. Contracts are also important in politics, not only because of "social contract" theories but also because contract enforcement has traditionally been considered a basic function of capitalist governments.

Whether enforced by a government, or otherwise, the contract is the basic building block of a free market economy. Over many centuries of cultural evolution has emerged both the concept of contract and principles related to it, encoded into common law. Algorithmic information theory suggests that such evolved structures are often prohibitively costly to recompute. If we started from scratch, using reason and experience, it could take many centuries to redevelop sophisticated ideas like property rights that make the modern free market work [Hayek].

The success of the common law of contracts, combined with the high cost of replacing it, makes it worthwhile to both preserve and to make use of these principles where appropriate. Yet, the digital revolution is radically changing the kinds of relationships we can have. What parts of our hard-won legal tradition will still be valuable in the cyberspace era? What is the best way to apply these common law principles to the design of our on-line relationships?

Computers make possible the running of algorithms heretofore prohibitively costly, and networks the quicker transmission of larger and more sophsiticated messages. Furthermore, computer scientists and cryptographers have recently discovered many new and quite interesting algorithms. Combining these messages and algorithms makes possible a wide variety of new protocols.

New institutions, and new ways to formalize the relationships that make up these institutions, are now made possible by the digital revolution. I call these new contracts "smart", because they are far more functional than their inanimate paper-based ancestors. No use of artificial intelligence is implied. A smart contract is a set of promises, specified in digital form, including protocols within which the parties perform on these promises.

Contracts Embedded in the World

The basic idea of smart contracts is that many kinds of contractual clauses (such as liens, 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 beginner's level problem in design with finite automata, dispense change and product fairly. 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, proactively enforced form, and provide much better observation and verification where proactive measures must fall short. And where the vending machine, like electronic mail, implements an asynchronous protocol between the vending company and the customer, some smart contracts entail multiple synchronous steps between two or more parties.

Other forerunners of smart contracts include POS (Point of Sale) terminals and cards, EDI (Electronic Data Interchange, used for ordering and other transactionsbetween large corporations), and the SWIFT, ACH, and FedWire networks for transferring and clearing payments between banks. These implement commercial security models, but too often with little heed paid to the contractual needs and obligations of the parties.

Attacks against Smart Contracts

A broad statement of the key idea of smart contracts, then, is to say that contracts should be *embedded in the world*. The mechanisms of the world should be structured in such a way as to make the contracts

(a) robust against naive vandalism, and

(b) robust against sophisticated, incentive compatible (rational) breach

A vandal can be a strategy or sub-strategy of a game whose utility is at least partially a function of one's own negative utility; or it can be a mistake by a contracting party to the same effect. "Naive" simply refers to both lack of forethought as to the consequences of a breach, as well as the relatively low amount of resources expended to enable that breach. Naive vandalism is common enough that it must be taken into consideration. A third category, (c) sophisticated vandalism (where the vandals can and are willing to sacrifice substantial resources), for example a military attack by third parties, is of a special and difficult kind that doesn't often arise in typical contracting, so that we can place it in a separate category and ignore it here. The distinction between naive and sophisticated strategies has been computationally formalized in algorithmic information theory

Some Basic Principapls of Contract Design

The threat of physical force is an obvious way to embed a contract in the world -- have a judicial system decide what physical steps are to be taken out by an enforcement agency (including arrest, confiscation of property, etc.) in response to a breach of contract. It is what I call a *reactive* form of security. The need to invoke reactive security can be minimized, but not eliminated, by making contractual arrangements *verifiable*, for example by recording a breach on a video camera, or putting a signature on a contract, in order to prove breach claims in court. *Observation* of a contract in progress, in order to detect the first sign of breach and minimize losses, also is a reactive form of security. A *proactive* form of security is a physical mechanism that makes breach expensive, such as a combination lock that makes access to a room containing trade secrets expensive without explicit authorization.

From common law, economic theory, and contractual conditions often found in practice, we can distill four basic objectives of contract design. The first of these is *observability*, the ability of the principals to observe each others' performance of the contract, or to prove their performance to other principals. The field of accounting is, roughly speaking, primarily concerned with making contracts an organization is involved in more observable.

A second objective *verifiability*, the ability of a principal to prove to an arbitrator that a contract has been performed or breached, or the ability of the arbitrator to find this out by other means. The disciplines of auditing and investigation roughly correspond with verification of contract performance. Observability and verifiability can also include the ability to differentiate between intentional violations of the contract and good faith errors.

A third objective of contract design is *privity*, the principle that knowledge and control over the contents and performance of a contract should be distributed among parties only as much as is necessary for the performance of that contract. This is a generalization of the common law principle of contract privity, which states that third parties, other than the designated arbitrators and intermediaries, should have no say in the enforcement of a contract. Generalized privity goes beyond this to formalize the common claim, "it's none of your business". Attacks against privity are epitomized by third parties Eve the eavesdropper, a passive observer of contents or performance, and malicious Mallet, who actively interferes with performance or steals service. Under this model privacy and confidentiality, or protecting the value of information about a contract, its parties, and its performance from Eve, is subsumed under privity, as are property rights. The field of security (especially, for smart contracts, computer and network security), roughly corresponds to the goal of privity.

A fourth objective is *enforceability*, and at the same time minimizing the need for enforcement. Improved verifiability often also helps meet this fourth objective. Reputation, built-in incentives, "self-enforcing" protocols, and verifiability can all play a strong part in meeting the fourth objective. Computer and network security also can contribute greatly to making smart contracts self-enforcing.

Smart contracts often involve trusted third parties, exemplified by an intermediary, who is involved in the performance, and an arbitrator, who is invoked to resolve disputes arising out of performance (or lack thereof). Privity implies that we want to minimize vulnerability to third parties. Verifiability and observabiliy often require that we invoke them. A mediator must be trusted with some of the contents and/or performance of the contract. An arbitrator must be trusted with some of the contents, and some of the history of performance, and to resolve disputes and invoke penalties fairly. In smart contract design we want to get the most out of intermediaries and arbitrators, while minimizing exposure to them. One common outcome is that confidentiality is violated only in case of dispute.

In the future the size distribution of multinational companies will approach that of local business, giving rise to multinational small business . Legal barriers are the most severe cost of doing business across many jurisdictions. Smart contracts can cut through this Gordian knot of jurisdictions. Where smart contracts can increase privity, they can decrease vulnerability to capricious jurisdictions. Where smart contracts can increase observability or verifiability, they can decrease dependence on these obscure local legal codes and enforcement traditions.

The consequences of smart contract design on contract law and economics, and on strategic contract drafting, (and vice versa), have been little explored. As well, I suspect the possibilities for greatly reducing the transaction costs of executing some kinds of contracts, and the opportunities for creating new kinds of businesses and social institutions based on smart contracts, are vast but little explored. The "cypherpunks" have explored the political impact of some of the new protocol building blocks. The field of Electronic Data Interchange (EDI), in which elements of traditional business transactions (invoices, receipts, etc.) are exchanged electronically, sometimes including encryption and digital signature capabilities, can be viewed as a primitive forerunner to smart contracts. Indeed those business forms can provide good starting points and channel markers for smart contract designers.

Observability & Hidden Actions

One important task of smart contracts, that has been largely overlooked by traditional EDI, is critical to "the meeting of the minds" that is at the heart of a contract: communicating the semantics of the protocols to the parties involved. There is ample opportunity in smart contracts for "smart fine print": actions taken by the software hidden from a party to the transaction. For example, grocery store POS machines don't tell customers whether or not their names are being linked to their purchases in a database. The clerks don't even know, and they've processed thousands of such transactions under their noses. Thus, via hidden action of the software, the customer is giving away information they might consider valuable or confidential, but the contract has been drafted, and transaction has been designed, in such a way as to hide those important parts of that transaction from the customer.

To properly communicate transaction semantics, we need good visual metaphors for the elements of the contract. These would hide the details of the protocol without surrendering control over the knowledge and execution of contract terms. A primitive but good example is provided by the SecureMosiac software from CommerceNet. Encryption is shown by putting the document in an envelope, and a digital signature by affixing a seal onto the document or envelope. On the other hand, Mosaic servers log connections, and sometimes even transactions, without warning users -- classic hidden actions.

Cryptographic Building Blocks

Protocols based on mathematics, called *cryptographic protocols*, tre the basic building blocks that implement the improved tradeoffs between observability, verifiability, privity, and enforceability in smart contracts. Contrary to the common wisdom, obscurity is often critical to security. Cryptographic protocols are built around foci of obscurity called *keys*. A key's immense unknown randomness allows the rest of the system to be simple and public. The obscurity of a large random number, so vast that a lucky guess is unlikely in in, if desired, the lifetime of the universe, is the foundation upon which cryptographic protocols, and in turn smart contracts, are built.

A wide variety of new cryptographic protocols have emerged in recent years. The most tradtional kind of cryptography is *secret key* cryptography, in which Alice and Bob (our exemplar parties to a smart contract) use a single shared, prearranged key to encrypt messages between them. A fundamental problem we will see throughout these protocols is the need to keep keys secret, and *pubilic key* cryptography helps solve this. In this technique, Alice generates two keys, called the private and public keys. She keeps the private key secret and well protected, and publishes the public key. When Bob wishes to send a message to Alice, he encrypts a message with her public key, sends the encrypted message, and she decrypts the message with her private key. The private key provides a "trapdoor" that allows Alice to compute an easy inverse of the encryption function that used the public key. The public key provides no clue as to what the private key is, even though they are mathematically related. The RSA algorithm is the most widely used method of public key cryptography.

Public key cryptography also makes possible a wide variety of *digital signatures*. These proves that a piece of data (hereafter referred to as just an "object") was in active contact with the private key corresponding to the signature: the object was actively "signed" with that key. The digital signature probably should have been called a "digital stamp" or "digital seal" since its function resembles more those methods than an autograph. In particular, it is not biometric like an autograph, although incorporation of a typed-in password as part of the private key used to sign can sometimes subsitute for an autograph. In many Asian countries, a hand-carved wooden block, called a "chop", is often used instead of autographs. Every chop is unique, and because of the unique carving and wood grain cannot be copied. A digital signature is similar to the chop, since every newly generated key is unique, but it is trivial to copy the key if obtained from the holder. A digital signature relies on the assumption that the holder will keep the private key secret.

A *blind signature* is a digital signature and secret-key encryption protocol that together have the mathematical property of commutativity, so that they can be stripped in reverse of the order they were applied. The effect is that Bob "signs" an object, for which he can verify its general form, but cannot see its specific content. Typically the key of the signature defines the meaning of the signed object, rather than the contents of the object signed, so that Bob doesn't sign a blank check. Blind signatures used in digital bearer instruments, where Bob is the clearing agent, and in Chaumian credentials , where Bob is the credential issuer.

Secret sharing is a method of splitting a key (and thus control over any object encrypted with that key) into N parts, of which only M are needed to recreate the key, but less than M of the parts provide no information about the key. Secret sharing is a potent tool for distributing control over objects between principals.

The zero-knowledge interactive proof is an alternative to public key methods for challenge-response identification. Otherwise normally functioning parties who have an incentive to respond properly to the challenge, but fail to do so, do not possess the key), without revealing any information about that privatge key to the challenger or any eavesdroppers. ZKIPs are currently used for authentication, and in smart weapons for Identification Friend or Foe (IFF).

Information about who is talking to whom, such as can be found on telephone bills, can be quite valuable even without records of the actual content. Confidential messsaging is necessary for the some of the privity features of Chaumian credentials and bearer securities to be strongly implemented on an actual network. To provide this traffic confidentiality, a *digital mix* can allow parties to communicate across a network without revealing their partners to network providers or the outside world. In a mix, traffic analysis by Eve is prevented by the Russian-doll encryption of the message by the sender with the public keys of each mix operator in the chain, and the mixing of messages by each operator, so that panoptic wiretapper Eve loses track of the messages. For the sender/recipient pair to remain confidential, only 1 out of N of the operators needs to be trusted with their local traffic information, although Eve can sometimes gather statistics over large numbers of messages between the same partners to eventually guess who is talking to whom. The communicating parties can also be mutually anonymous, and with normal encryption need trust no other parties with the content of messages. The "Mixmaster" software on the Internet implements most of the features of a digital mix[Mixmaster].

Protection of Keys

So far, we've assumed parties like Alice and Bob are monolithic. But in the world of smart contracts, they will use computer-based software agents and smart cards to do their electronic bidding. Keys are not necessarily tied to identities, and the task of doing such binding turns out to be more difficult than at first glance. Once keys are bound, they need to be well protected, but wide area network connections are notoriously to hacking.

If we assume that the attacker has the ability to intercept and redirect any messages in the network protocol, as is the case on wide area netowrks such as the Internet. then we must also assume, for practical all commercial operating systems, that they would also be able to invade client if not merchant computers and find any keys lying on the disk.

There's no completely satisfactory solution to end point operations security from network-based attacks, but here's a strategy for practically defanging this problem for public-key based systems:

All public key operation are are done inside an unreadable hardware board on a machine with a very narrow serial-line connection (ie, it carries only a simple single-use protocol with well-verified security) to a dedicated firewall. Such a board is available, for example, from Kryptor, and I believe Viacrypt may also have a PGP-compatable board. This is economical for central sites, but may be less practical for normal users. Besides better security, it has the added advantage that hardware speeds up the public key computations.

If Mallet's capability is to physically sieze the machine, a weaker form of key protection will suffice. The trick is to hold the keys in volatile memory. This makes the PC proof from physical attacks -- all that needed to destroy the keys is to turn off the PC. If the key backups can be hidden in a different, secure physical location, this allows the user of this PC to encrypt large amounts of data both on the PC itself and on public computer networks, without fear that physical attack against the PC will compromise that data. The data is still vulnerable to a "rubber hose attack" where the owner is coerced into revealing the hidden keys. Protection against rubber hose attacks might require some form of Shamir secret sharing which splits the keys between diverse phgsical sites.

The Man In the Middle & PGP's Web of Trust

How does Alice know she has Bob's key? Who, indeed, can be the parties to a smart contract? Can they be defined just by their keys? Do we need biometrics (such as autographs, typed-in passwords, retina scans, etc.)?

The public key cryptography software package "Pretty Good Privacy" (PGP) uses a model called "the web of trust". Alice chooses *introducers* whom she trusts to properly identify the map between other people and their public keys. PGP takes it from there, automatically validating any other keys that have been signed by Alice's designated introducers.

There are two entirely separate criteria PGP uses to judge a public key's usefulness:

1) Does the key actually belong to whom it appears to belong? In other words, has it been certified with a trusted signature?

2) Does it belong to an introducers, someone you can trust to certify other keys?

Having been told by Alice the answer to the second question, PGP can calculates the answer to the first question for the public keys Alice has collected.

Keys that have been certified by a trusted introducer are deemed valid by PGP. The keys belonging to trusted introducers must themselves be certified either by you or by other trusted introducers. This "transitivity" introduces an implicit third criterion

3) Does the key belong to someone you can trust to introduce other introducers?

PGP confuses this with criterion (2). It is not clear that any single person has enough judgement to properly undertake task (3), nor has a reasonable institution been proposed that will do so. This is one of the unsolved problems in smart contracts.

PGP also can be given trust ratings and programmed to compute a weighted score of validity-- for example, two marginally trusted signatures might be considered as credible as one fully trusted signature.

Any keys in Alice's secret key ring are "axiomatically" valid to Alice's PGP program, needing no introducer's signature. PGP also assumes that Alice ultimately trusts herself to certify other keys.

It is believed that PGP causes the emergence of a decentralized fault-tolerant web of confidence for all public keys, but a chain of introduced introducers grows weak very quickly, due to lack of transitivity.

PGP's grass-roots approach contrasts sharply with tradtional public key management schemes, such as X.509 and the related Privacy Enhanced Mail (PEM). These standard schemes subsistute a heirarchical system of introducers called certification authorities (CAs).

Notaries Public

Two different acts are often called "notarization". The first is simply where one swears to the truth of some affidavit before a notary or some other officer entitled to take oaths. This does not require the notary to know who the affiant is. The second act is when someone "acknowledges" before a notary that he has executed a document as ``his own act and deed." This second act requires the notary to know the person making the acknowledgment. Thus, for example, the form of an acknowledgment can go something like this:

On this ____ day of ___, 19__, personally appeared before me ____, known to me and known to me to be the person who signed the foregoing instrument, and acknowledged that he executed the same as his own act and deed.

In the first type of act of notarization, the notary merely certifies that the affiant swore the statement was true. In the second type the notary actually vouches that the person making the acknowledgment was who he claims to be.

Problems with Certification

These roles of a notary public are substantially different from the alleged role of heirarchies of "certification authorities" (CAs) in PEM/X.509, and the "web of trust" in PGP, to "prove identity". In fact the certificates generated by these systems do no such thing. Rather a certficate *proves that a claim was made* by a CA at some time in the past. The (implicit) claim is that a particular key belonged to a particular person at that time. That key is not biometric like an autograph, and can thus be transferred at any time. Furthermore, false claims can be made by a CA about what keys an end-user has held, and the end-user can be stuck with no evidence of CA fraud; nor does the CA have any way of proving that their claim is correct if an end-user challenges it. It is extremely difficult, on the other hand, for notary publics to forge autographs and expect to get away with it often enough to maintain their professional reputations.

Both the PGP web of trust and X.509 models suffer from more flaws. A single rooted heirarchy assumes a mythological beast called a universally trusted entity. Heirarchies in general create rigid structures that don't fit the way knowledge about keys and keyholders, and incentives to accurately reflect that knowledge, are distributed among people in the real world. In turn, hile PGP distinguishes between "Alice holds a key" and "Alice can be trusted to certify a key", it does not follow that the second claim that Alice can be trusted to validate another issuer. There is little transitivity: seeing Alice's key certified by Bob, whom I know and trust, tells me little about whether Alice's certification of Charlie's key is correct. Seeing Alice certified by Bob as an introducer tells me little about wether Alice can be trusted to certify other introducers. It does tell me that I know to blame Alice if her claim turns out to be wrong; although it's far from clear that Alice has any legal liability.

There is an even more severe flaw when public key is used for digital signatures. Because the claim was only made in the past, both PGP and X.509 allow the end user to plausible deny that they "signed" a document; and conversely if one's key is surreptitiously stolen, or for other reasons no revocation action is taken, there is no way to prove that one did not "sign" the documenent digitally "signed" with the stolen key. Finally, there is no widely accepted legal agreement on what is specifically being claimed when one "certifies" a key, nor is there any built-in or widely used mechanism for describing the actual claim one is making. Real world CAs have a nasty habit of disclaiming liability for their mistakes, or for the misunderstandings that will arise out of the often vague and sloppy, sometimes implicit claims they make. Finally, there are a wide variety of other claims one might make about a key, such as "this key belongs to an office", "this key belongs to a server", "this keyholder has a good credit rating", "use this key to decrypt your new copy of Microsquish Expel", "this key is good for 100 MB of downloads from our web server", etc. which are facilitated by neither X.509 nor the PGP web of trust.

We know too little about the best uses of public-key cryptography to establish such fixed methods with such narrow semantics. This author has suggested a mechanism which modifies the PGP web of trust to create arbitrary certificate with a form roughly as follows:

Key about which a claim is being made type of claim, in some standard one-line format (like MIME types) Plain text description of claim Timestamp digitally "signed", Alice's key

In other words, all claims about keys should be *explicit*, readily known from reading the certificate itself, and no kind of claim should be arbitrarily excluded by the mechanism. The legal force of the claim can be based on the text itself, rather than overstated, obscure, and often implicit interpretations about what "certifying" is supposed to mean. Standard kinds of claims will emerge, including perhaps more transitive "good judge of judgement" certificates for which chain-following software can be written, and non-transitive "is-a-person" credentials directly "bound" to traditional notarized identification by a physical notarization protocol that includes both autographs and digital "signatures". More likely, new and more useful kinds of certificates will evolve. These standards should be allowed to emerge out of the wide varieties of possible end uses, much like case law has matured over time, rather than being dictated by our current very inexperienced understanding.

Meanwhile, there is a more practical defense aggainst the man in the middle attack: advertise, early and often. Users of an insecure network can communicate the integrity of a key reasonably well by tying it to a persistent pattern of behavior: for example posts in a persistent style from a persistent e-mail address, persistence of a key unchallenged on a key server, etc. This is the most practical and widely used way by which PGP users gain confidence in public keys, and it does not require certification authorities or introducers. Those who advertise their keys widely, and those who are well known, are more likely to have keys bound to their person.

Virtual Personae

"Identity" is hardly the only thing we might want map to a key. After all, physical keys we use for our house, car, etc. are not necessarily tied to our identity -- we can loan them to trusted friends and relatives, make copies of them, etc. Indeed, in cyberspace we might create "virtual personae" to reflect such multi-person relationships, or in contrast to reflect different parts of our personality that we do not want others to link. Here is a possible classification scheme for virtual personae, pedagogically presented:

A *nym* is an identifier that links only a small amount of related information about a person, usually that information deemed by the nym holder to be relevant to a particular organization or community. Examples of nyms include electronic bulletin board nicknames, pen names, aliases, and brand names. A nym may gain reputation within its community. For example, a conglomerate may sell a wide variety of brand names, each reputable in its own market niche. With Chaumian credentials , a nym can take advantage of the positive credentials of the holder's other nyms, as provably linked by the is-a-person credential.

A *true name* is an identifier that links many different kinds of information about an person, such as a full birth name or social security number. As in magick, knowing a true name can confer tremendous power to one's enemies. It also can have major economic value among those who cooperate peacefully, as in the use of direct marketing to target product information to those persons most likely to be interested in those particular products.

A *persona* is any perstient pattern of behavior, along with consistently grouped information such as key(s), name(s), network address(es), writing style, and services provided.

A *reputable name* is a nym or true name that has a good reputation, usually because it carries many positive credentials, has a good credit rating, or is otherwise highly regarded. Companies strive to carry reputable brand names, while professionals such as doctors and lawyers strive to have many good personal recommendations of their name. Reputable names can be difficult to transfer between parties, because reputation assumes persistence of behavior, but such transfer can sometimes occur (for example, the sale of brand names between companies).

Constructing Smart Contracts

Blind signatures can be used to construct *digital bearer instruments*, objects identified by a unique key, and issued, cleared, and redeemed by a clearing agent. When an object is transferred, the trasferee can request the clearing agent to verify that the key has never before been cleared, and issue a new key. The clearing agent prevents multiple clearing of particular objects, but can be prevented from linking particular objects one or both of the clearing nyms who transferred that object. These instruments come in an "online" variety, cleared during every transfer, and thus both verifiable and observable, and an "offline" variety, which can be transfered without being cleared, but is only verifiable when finally cleared, by revealing any the clearing nym of any intermediate holder who transfered the object multiple times (a breach of contract). Privacy from the clearing agent can take the form of transferee-unlinkability, transferer-unlinkability, or "double blinded" where both transferer and transferee are unlinkable by the clearing agent.

Digital cash is the premier example of a digital bearer instrument, in which the clearing agent is a bank. Bearer instrument protocols enable online payment while honoring the characteristics desired of bearer notes, especially unforgeability (via the clearing mechanism) and transfer confidentiality (via blinding).

To implement a full transaction of payment for services, we need more than just the digital cash protocol; we need a protocol that guarantees that service will be rendered if payment is made, and vice versa. Current commercial systems use a wide variety of techniques to accomplish this, such as certified mail, face to face exchange, reliance on credit history and collection agencies to extend credit, etc. Smart contracts have the potential to greatly reduce the fraud and enforcement costs of many commercial transactions.

A *credential* is a claim made by one party about another. A *positive credential* is one the second party would prefer to reveal, such as a degree from a prestigious school, while that party would prefer not to reveal a *negative credential* such as a bad credit rating.

A Chaumian credential is a cryptographic protocol for proving one possesses claims made about onself by other nyms, without revealing linkages between those nyms. It's based around the *is-a-person credential* the true name credential, used to prove the linkage of otherwise unlinkable nyms, and to prevent the transfer of nyms between parties.

Another form of credential is *bearer credential*, a digital bearer instrument where the object is a credential. Here the second party in the claim refers to any bearer -- the claim is tied only to the reputable name of issuing organization, not to the nym or true name of the party holding the credential.

Smart Property

We can extend the concept of smart contracts to property. Smart property might be created by embedding smart contracts in physical objects. These embedded protocols would automatically give control of the keys for operating the property to the party who rightfully owns that property, based on the terms of the contract. For example, a car might be rendered inoperable unless the proper challenge-response protocol is completed with its rightful owner, preventing theft. If a loan was taken out to buy that car, and the owner failed to make payments, the smart contract could automatically invoke a lien, which returns control of the car keys to the bank. This "smart lien" might be much cheaper and more effective than a repo man. Also needed is a protocol to provably remove the lien when the loan has been paid off, as well as hardship and operational exceptions. For example, it would be rude to revoke operation of the car while it's doing 75 down the freeway.

Smart property is software or physical devices with the desired characteristics of ownership embedded into them; for example devices that can be rendered of far less value to parties who lack possesion of a key, as demonstrated via a zero knowledge interactive proof.

One method of implementing smart property is thru operation necessary data (OND): data necessary to the operation of smart property. For example, a complex, OND can be proprietary firing sequence needed to operate a computerized engine, a CAM file needed to manufacture a specialized part, etc. To avoid theft of service, ZKIP is required to open an encrypted channel to the device. To avoid leaking the OND to Eve, tamper detection combined with a dead-man switch can be used on the device end of the channel.

We might also usedand engrained immobilizing or destructive devices to foil attempts to hot-wire smart property.

A *smart lien* is the sharing of a smart property between parties, usually two parties called the owner and the lienholder. This property may be in the proximate possesion of the owner or the lienholder, corresponding to the common-law notions of "artisan's lien" and "innkeeper's lien" respectively. Smart liens might be used to secure lines of credit, insurance policies,and many other kinds of contracts that involve smart property.

How can debts be collected? No wise bank will lend unless the lendee can either be coerced into repaying the debt, or the loan is more than covered by securely liened collateral plus some conservative function of its reputation for payment in full and on time. For all parties, both credit and liability are closely related, and limited.

The liability of a party is limited by that party's liens and by the ability to deter that party by threatening punishment for violating contracts (ie, committing crimes as defined by the contract with the jurisdiction). The potential for other actions an party might take that cause liability, such as damage to others' persons or property, also need to be limited. More on that later.

Many parties, especially new entrants, may lack this reputation capital, and will thus need to be able to share their property with the bank via secure liens. A lien is, in a practical sense, a method of sharing a piece of property between the "owner of record" and a "lienholder", instead of the property having strictly one owner. Liens are used in many large credit transactions, such as auto loans, mortgages, farm loans, etc. They are enforced by the jurisdiction specified in the contract; usually this enforcement is done by the government and subsidized by the taxpayers rather than paid for by the contracting parties. (In fact this usually is the case with contracts and property rights in general, the enforcement clause is an implicit government subsidy). One way to implement a lien without governments is via co-signing with your privately chosen arbitrator (as long as the arbitratror has a good reputation and the contractual right to take appropriate action against you). Smart liens might greatly expand the privity and security of such arrangements.

As is the case today, credit problems will usually be solved by artfully written, menacing dunning letters and dings to one's credit rating long before the lien needs to be invoked. However, the lien needs to be enforceable to make these dunning letters credible over the long run.

What about extending the concept of contract to cover agreement to a prearranged set of tort laws? These tort laws would be defined by contracts between private arbitration and enforcement agencies, while customers would have a choice of jurisdictions in this system of free-market "governments". If these privately practiced law organizations (PPLs for short) bear ultimate responsibility for the criminal activities of their customers, or need to insure lack of defection or future payments on the part of customers, they may in turn ask for liens against their customers, either in with contractual terms allowing arrest of customers under certain conditions (eg if they commit acts specified as criminal by the PPL contract) or (more likely for mobile world-traveling and virtual pseudonymous customers) smart liens against liquid assets such as bank accounts and investment portfolios. Smart liens over information, such as digital bearer securities, can be implemented via secret sharing (two or more keys required to unlock the encryption).

Other important areas of liability include consumer liability and property damage (including pollution). There need to mechanisms so that, for example, pollution damage to others' persons or property can be assessed, and liens should exist so that the polluter can be properly charged and the victims paid. Where pollution is quantifiable, as with SO2 emissions, markets can be set up to trade emission rights. The PPLs would have liens in place to monitor their customer's emissions and assess fees where emission rights have been exceeded.

Alas, there are some dangers where maximum damage could far surpass any liens. A good rule of thumb here is that if the risk is against a third party, and it cannot be liened or insured against, then PPLs should not allow it to be taken. PPLs that allow their customers to take such risks against non-PPL parties would ruin their credit rating. One example of such a risk is building a nuclear plant for which no insurance company is willing to submit liability coverage. If a plant is safe, presumably one should be able to convince a good insurance company to cover its potential to damage others' property.

Conclusion

Digital cash is here today, and many more smart contract mechanisms are being designed. So far the design criteria important for automating contract execution have come from disparate fields like economics and cryptography, with little cross-communication: little awareness of the technology on the one hand, and little awareness of its best business uses other. The idea of smart contracts is to recognize that these efforts are striving after common objectives, which converge on the concept of smart contracts.

References:

Oliver Williamson, *The Economic Institutions of Capitalism*
Janet Landa, *Trust, Ethnicity, and Identity*
The New Palgrave: Allocation, Information, and Markets
Bruce Schneier, *Applied Cryptography*
Crypto and *Eurocrypt* conference proceedings, 1982-1994.
"Crypto Rebels", Wired #2, also cypherpunks mailing list
(mail to majordomo@toad.com with body "subscribe cypherpunks")
Perry H. Beaumont, *Fixed Income Synthetic Assets*
Frederich Hayek, *The Fatal Conceit*
Ming Li & Paul Vitanyi, *An Introduction to Kolmogorov Complexity
and Its Applications*
Vernor Vinge, "True Names" (fiction), from *True Names and Other Dangers*
Mixmaster (ptr to web site)
PGP (ptr to web site)
Tim May, "Cyphernomicon"
David Chaum, "Security Without Identification"
David Chaum, "Achieving Electronic Privacy"

Agorics web site
DigiCash web site