Article Economics Today

06/11/2018

17 Blockchain Applications That Are Transforming Society And Blockcain Analysis

Kittinant Phitsuwan

กิตตินันต์ พิศสุวรรณ

Harvard Business School

The early internet dealt with intangibles. You sent or received emails, corresponded on forums, read and distributed articles. This modern internet deals with assets, your most valuable immediate items that you can touch and want to protect. These assets are stored in encoded form on a network-to-network chain called the blockchain or ledger, where each participant sees who you do business with. This not only protects your business dealings and prevents theft, but, also, simplifies your affairs, quickens the process, reduces errors, and saves you from hiring a third party.

This decentralized blockchain system is going to change your life from the way you transact business or manage assets, to the way you use your machines, vote, rent a car, and even prove who you are. Along the way, it will transform banks and other financial institutions, hospitals, companies, and governments among others.

If you are looking to create your own applications, please check out our blockchain courses.

Here's what the blockchain does and what it means to you.

Blockchain Finance

Decentralized cryptocurrencies.

At its simplest, cryptocurrencies, or digital coins, are coins that are passed through an electronic network. You can make transactions by check, wiring, or cash. You can also use a type of virtual currency, most famously Bitcoin (BTC) but also Litecoin, Peercoin, or Dogecoin, among others, where you use an electronic coded address to make the transaction.

Train to Become A Blockchain Developer

Start Your Free Trial Today!

The more valuable the transaction, the more you want to protect it. Traditional systems hire a mediator, such as a banker or a remittance company to ensure trust. Islanders of Yap had a different solution. They kept a mental record of who owned what and referred to this distributed community record when disputes arose. The blockchain is this community record on a wider, digital scale. It extends across the globe, with computer users from Yemen, Rome, Vermont and so forth where each node in the network records and verifies the data

of each transaction that occurs within the network. Records are permanent, comprehensive and public – which is why users love the blockchain for finagling questionable or risky transactions.

How it works

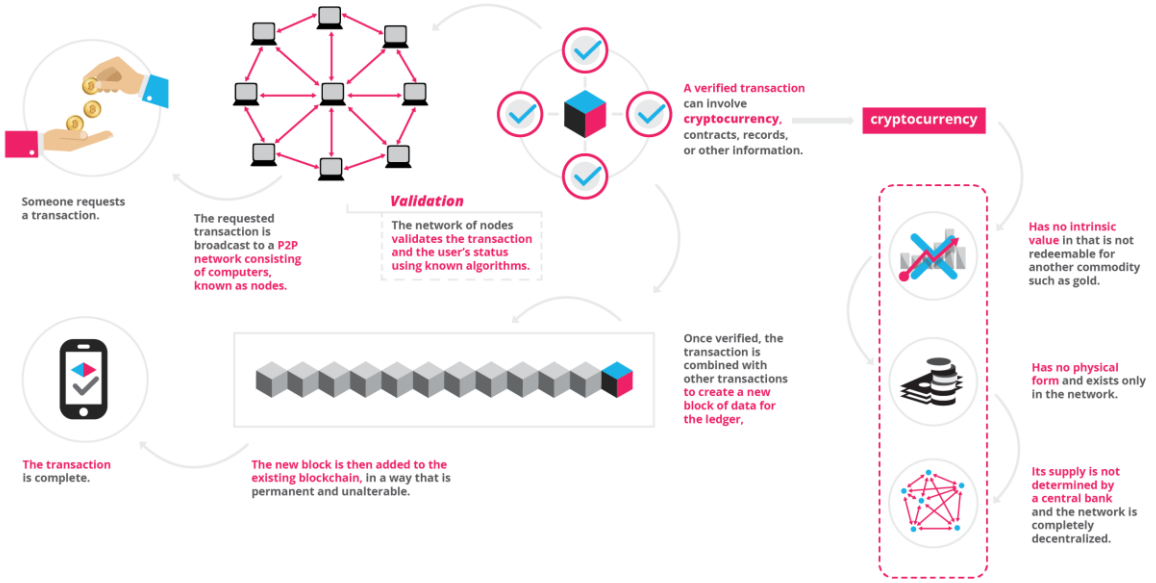
Each transaction is a digital ‘block’ that needs to be verified before it’s allowed to enter the system.

Questions include:

- Is the money there?
- Are sender and receiver reputable?
- Is the request legitimate? And so forth.

Each computer on the network competes on unscrambling the answers, and the winning computer adds this ‘block’ to the ‘blockchain’ in the order that the ‘block’ arrived. The winner broadcasts his proof to the rest of the network, which checks that proof and verifies it before queuing the ‘block’ to complete the transaction. Parties involved are assured that participants have screened and okayed the transaction.

The process not only cuts down on fraud, such as double spending or spams, but also transfers funds simply, safely, and fast.



17 Blockchain Applications That Are Transforming Society

Blockchain Business

Financial Services

Traditional systems tend to be cumbersome, error-prone and maddeningly slow. Intermediaries are often needed to mediate the process and resolve conflicts. Naturally, this costs stress, time, and money. In contrast, users find the blockchain cheaper, more transparent, and more effective. Small wonder that a growing number of financial services are using this system to introduce innovations, such as smart bonds and smart contracts. The former automatically pays bondholders their coupons once certain preprogrammed terms are met. The latter are digital contracts that self-execute and self-maintain, again when terms are met.

Examples of blockchain financial services

Asset Management: Trade Processing and Settlement

Traditional trade processes within asset management (where parties trade and manage assets) can be expensive and risky, particularly when it comes to cross-border transactions.

Each party in the process, such as broker, custodian, or the settlement manager, keeps their own records which create significant inefficiencies and room for error. The blockchain ledger reduces error by encrypting the records. At the same time, the ledger simplifies the process, while canceling the need for intermediaries.

Insurance: Claims processing

Claims processing can be a frustrating and thankless procedure. Insurance processors have to wade through fraudulent claims, fragmented data sources, or abandoned policies for users to state a few – and process these forms manually. Room for error is huge. The blockchain provides a perfect system for risk-free management and transparency. Its encryption properties allow insurers to capture the ownership of assets to be insured.

Payments: Cross-Border Payments

The global payments sector is error-prone, costly, and open to money laundering. It takes days if not longer for money to cross the world. The blockchain is already providing solutions with remittance companies such as Abra, Align Commerce and Bitspark that offer end-to-end blockchain powered remittance services. In 2004, Santander became one of the first banks to merge blockchain to a payments app, enabling customers to make international payments 24 hours a day, while clearing the next day.

Smart Property

A tangible or intangible property, such as cars, houses, or cookers, on the one hand, or patents, property titles, or company shares, on the other, can have smart technology embedded in them. Such registration can be stored on the ledger along with contractual details of others who are allowed ownership in this property. Smart keys could be used to facilitate access to the permitted party. The ledger stores and allows the exchange of these smart keys once the contract is verified.

The decentralized ledger also becomes a system for recording and managing property rights as well as enabling the smart contracts to be duplicated if records or the smart key is lost.

Making property smart decreases your risks of running into fraud, mediation fees, and questionable business situations. At the same time, it increases trust and efficiency.

Examples of Blockchain Smart Property.

Unconventional money lenders/ hard money lending

Smart contracts can revolutionize the traditional lending system. For instance, unconventional money lenders (e.g. hard money lenders) service borrowers who have poor credit with needed loans – while charging two to ten percent of the loan amount and claiming their property as collateral. Too many borrowers fall into bankruptcy and lose homes. The blockchain can undercut this by allowing a stranger to loan you money and taking your smart property as collateral. No need to show the lender credit or work history. No need to manually process the numerous documents. The property's encoded on the blockchain for all to see.

Your car/ smartphone

Primitive forms of smart property exist. Your car-key, for instance, may be outfitted with an immobilizer, where the car can only be activated once you tap the right protocol on the key. Your smartphone too will only function once you type in the right PIN code. Both work on cryptography to protect your ownership.

The problem with primitive forms of smart property is that the key is usually held in a physical container, such as the car key or SIM card, and can't be easily transferred or copied. The blockchain ledger solves this problem by allowing blockchain miners to replace and replicate a lost protocol.

Blockchain Internet-of-Things (IoT)

Any material object is a 'thing.' It becomes an internet of things (IoT) when it has an on/ off switch that connects it to the internet and to each other. By being connected to a computer network, the object, such as a car, become more than just an object. It is now people-people, people-things, and things-things. The analyst firm Gartner says that by 2020 there will be over 26 billion connected devices. Others raise that number to over 100!

How does the IoT affect you? Your printer can automatically order cartridges from Amazon when it runs low. Your alarm clock will change your time for brewing coffee, while your oven will produce an immaculately timed turkey for Thanksgiving. These are just some examples. On a larger scale, cities and governments can use IoT to develop cleaner environments, more efficient energy use and so-called 'smart cities,' to improve how we live and work.

Where the blockchain comes in

As in all cases, the blockchain ledger provides security to this Internet of things. With billions of devices linked together, cybersecurity experts worry how to make sure this distributed information stays secure.

What can companies do to protect their systems from being invaded?

- How can inventors shield their ideas?
- How should governments protect their secret information from spies and potential terrorists?

Then, there's the problem of how to organize and analyze this massive amount of data that's coming from these related devices.

Enter the blockchain ledger system that ensures that information is only accepted and released to trusted parties. The ledger grants parties a management platform for analyzing the vast amounts of data.

Examples of Blockchain Internet-of-Things (IoT)

- **Smart Appliances**

A smart appliance is a device that connects to the internet and gives you more information and control than before. For instance, a code connected to your appliance can be linked to the internet and alert you when your cookies are ready or if your laundry has stopped.

These alerts keep your appliances in good condition, they save you money regarding energy efficiency and help you control your devices when away from home, among other benefits. Encrypting these appliances on the blockchain protects your ownership and enables transferability.

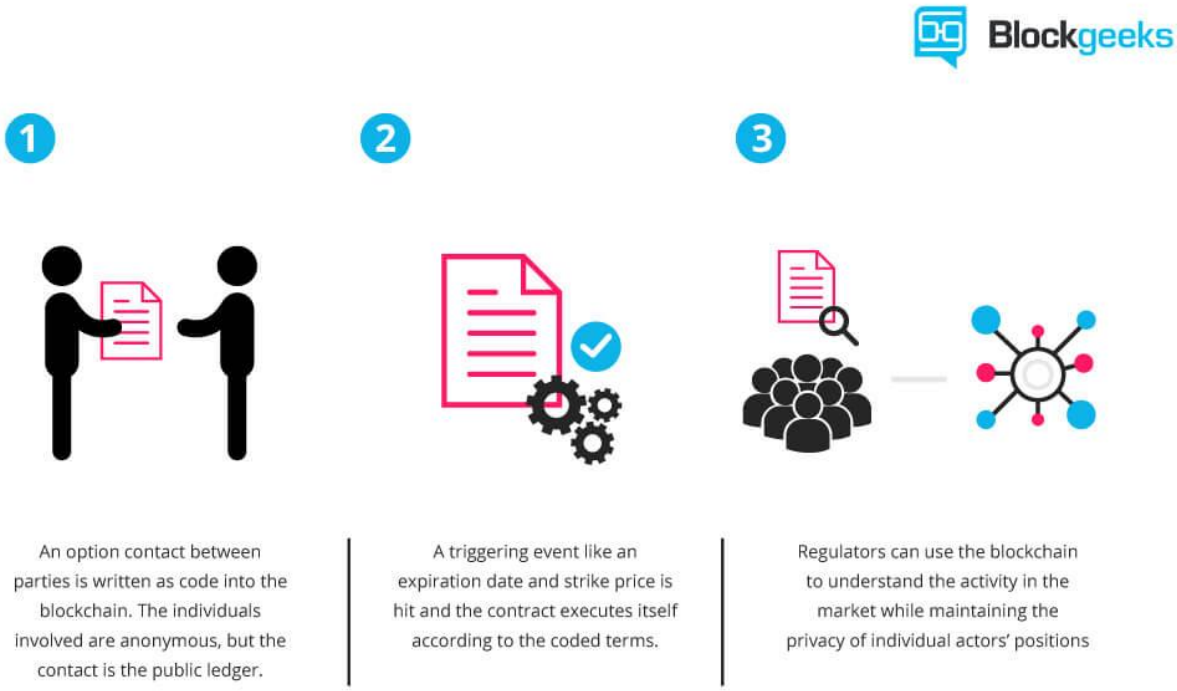
- **Supply Chain Sensors**

Sensors give companies end-to-end visibility of their supply chain by providing data on the location and condition of the supplies as they are transported around the globe. As of 2016, a Deloitte and MHI report surveyed 99 leading supply chain companies and found that sensors were used by 44% of these respondents. Eighty-seven percent of these industries said they plan to use the technology by 2020. The technology is expected to grow to 1 trillion by 2022 and to 10 trillion sensors by 2030, according to this sme Deloitte and MHI report. The blockchain stores, manages, protects and transfers this smart information.

Smart Contracts

Smart contracts are digital which are embedded with an if-this-then-that (IFTTT) code, which gives them self-execution. In real life, an intermediary ensures that all parties follow through on terms. The blockchain not only waives the need for third parties, but also ensures that all ledger participants know the contract details and that contractual terms implement automatically once conditions are met.

You can use smart contracts for all sort of situations, such as financial derivatives, insurance premiums, property law, and crowd funding agreements, among others.



Examples of Blockchain Smart Contracts

Blockchain Healthcare

Personal health records could be encoded and stored on the blockchain with a private key which would grant access only to specific individuals. The same strategy could be used to ensure that research is conducted via HIPAA laws (in a secure and confidential way). Receipts of surgeries could be stored on a blockchain and automatically sent to insurance providers as proof-of-delivery. The ledger, too, could be used for general health care management, such as supervising drugs, regulation compliance, testing results, and managing healthcare supplies.

Blockchain music

Key problems in the music industry include ownership rights, royalty distribution, and transparency. The digital music industry focuses on monetizing productions, while ownership rights are often overlooked. The blockchain and smart contracts technology can circuit this

problem by creating a comprehensive and accurate decentralized database of music rights. At the same time, the ledger and provide transparent transmission of artist royalties and real time distributions to all involved with the labels. Players would be paid with digital currency according to the specified terms of the contract.

- **Blockchain Government**

In the 2016 election, Democrats and Republicans questioned the security of the voting system. The Green Party called for a recount in Wisconsin, Pennsylvania, and Michigan. Computer scientists say hackers can rig the electronic system to manipulate votes. The ledger would prevent this since votes become encrypted. Private individuals can confirm that their votes were counted and confirm who they voted for. The system saves money, by the way, for the government, too.

The blockchain ledger, also, provides a platform for what we call "responsive, open data." According to a 2013 report from McKinsey and Company, open data – freely accessible government-sourced data that is available over the internet to all citizens – can make the world richer by $2.6 trillion. Startups can use this data to uncover fraudulent schemes, farmers can use it to perform precision farm-cropping, and parents can investigate the side effects of medicine for their sick children. Right now, this data is released only once a year and is, largely, non-responsive to citizens input. The blockchain, as a public ledger, can open this data to citizens whenever and wherever they want.

Examples of Blockchain Government

Public value/ community

The blockchain can facilitate self-organization by providing a self-management platform for companies, NGOs, foundations, government agencies, academics, and individual citizens. Parties can interact and exchange information on a global and transparent scale – think of Google Cloud, but larger and less risky.

Vested responsibility

Smart contracts can ensure that electorates can be elected by the people for the people so that government is what it's meant to be. The contracts specify the electorate's expectations and electors will get paid only once they do what the electorate demanded rather than what funders desired.

Blockchain Identity

Whether we like it or not, online companies know all about us. Some companies whom we purchase from sell our identity details to advertisers who send you their ads. The blockchain blocks this by creating a protected data point where you encrypt only the information that you want relevant people to know at certain times. For example, if you're going to a bar, the bartender simply needs the information that tells him you're over 21.

The blockchain protects your identity by encrypting it and securing it from spammers and marketing schemes.

Examples of Blockchain Identity:

Passports

The first digital passport launched on Github in 2014 and could help owners identify themselves online and off. How does it work? You take a picture of yourself, stamp it with a public and private key, both of which are encoded to prove it is legitimate. The passport is stored on the ledger, given a Bitcoin address with a public IP, and confirmed by Blockchain users.

Birth, wedding, and death certificates

Few things are more important than documents showing you're born, married, died which open your rights to all sorts of privileges (such as voting, working, citizenship), yet mismanagement is rife. Up to a third of children under the age of five have not been issued a birth certificate, the UNICEF reported in 2013. The blockchain could make record-keeping

more reliable by encrypting birth and death certification and empowering citizens to access this crucial information.

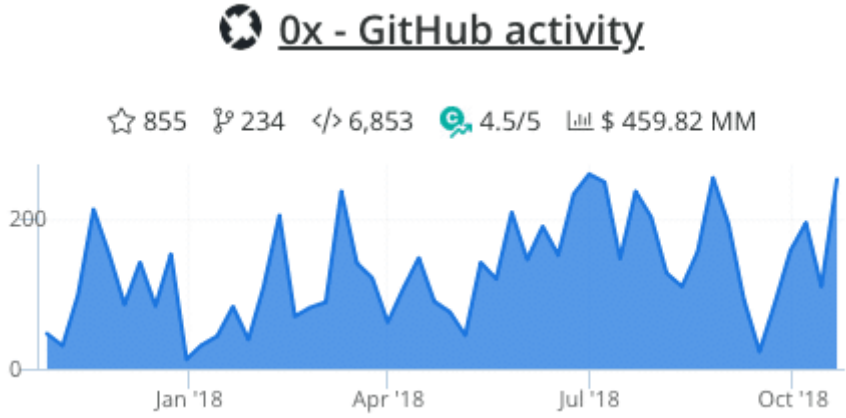
Personal Identification

We carry a range of identifications: Our driver's license, computer password, identity cards, keys, social security ID, and so forth. Blockchain ID is a digital form of ID that's engineered to replace all these forms of physical identification. In the future, fintech scientists say you'll be able to use the one digital ID for signing up at any registrar. It is open source, secured by the blockchain, and protected by a ledger of transparent account.

`We have a had a pretty interesting week in the cryptoverse. From Monero users benefitting from the hard fork to Australia taking giant strides towards wider crypto adoption. Let's take a look at the highlights from this last week in crypto.

A weekly check ofGitHub commits is a healthy indicator to know which projects are undergoing the most development. Checking this data is a good sign of knowing which projects are attracting developer attention. The following data and graphs have been takenfrom **Coin Checkup.**

#1 0x Project

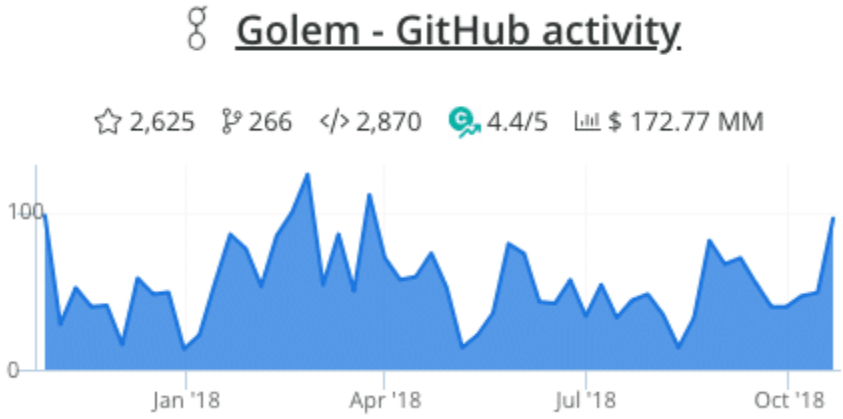


According to their website, "0x is an open, permissionless protocol allowing for ERC20 tokens [and potential other tokens] to be traded on the Ethereum blockchain" and will be used for "powering decentralized exchange."

0x came about in October 2016. Co-founders Will Warren and Amir Bandeali had a vision of a future where all kinds of assets, stocks, currencies, precious metals, could be traded publicly on the blockchain as tokens.

Last week they were at 110 commits and this week they managed to do a whopping 253 commits. So, nearly 143 commits were done this week alone. This shows that more and more developers are attracted by the 0x Project. They have seen quite a lot of developer activity since they completed mainnet testing for their v2.0 on September 24.

#2 Golem



Golem is one of the most ambitious projects being built on top of Ethereum. The idea is to build a marketplace for computational power and resources. Golem is headed by CEO Julian Zawistowski and raised ~$8 million in their ICO.

Last week they had 49 GitHub commits and they nearly doubled that this week with 96.

#3 Genesis Vision

As they move closer to their launch date (end of October), it looks like the developer activity on Genesis Vision is growing strength to strength.

Genesis Vision is the decentralized platform for the private trust management market, built on Blockchain technology and Smart Contracts. They combine exchanges, brokers, traders, and investors into a decentralized open and honest network, making the financial market even more global.

Last week they were at 81 commits and this week they have gone up to 115 commits.

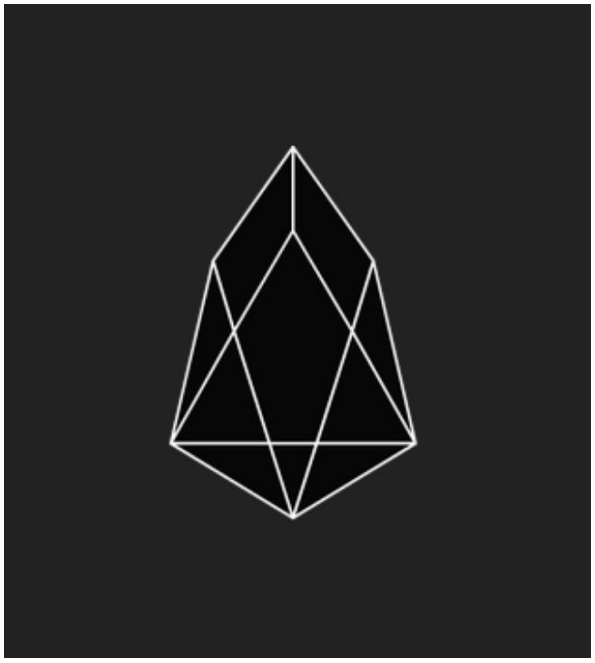
Must Watch Podcast Episodes Of The Week

Let's take a look at the top 3 podcast episodes of the week.

#1 Monica Quaintance at Epicenter

Monica Quaintance is the Head of Engineering and Adoption at Kadena. Kadena is building both a public network protocol and private blockchain infrastructure. Their Chainweb protocol will soon launch as a public network and smart contract platform.

Ultimate Guide to EOS Smart Contract Security. The crypto community became skeptical when the World's biggest ICO, EOS launched in June 2018 and it got freezed out for 2 days due to a software bug. But fast forward 4 months and EOS today accounts for more than double the transactions that Ethereum does today. Through the promise of free and faster transactions, the topmost Dapp of EOS has about 13,000 daily active users compared to just 2,000 of Ethereum's topmost Dapp.



EOS Smart Contract Security

By Rohan Agarwal

Some general smart contract vulnerabilities are applicable for almost all of the platforms. Like Ethereum, smart contracts written on EOS needs to be audited before going live on the mainnet. Fatal bugs in the contract can get exploited when the contracts are not battletested enough. In this guide, we will help you avoid the common pitfalls on your way to make the next killer dApp on EOS.

Before you read the guide, it is important to know about some prerequisite information regarding EOS development that will come handy while you read the guide. Knowledge of C++ is a must. The best place to start with the smart contract development is EOSIO's own documentation

Dealing with the ABI Dispatcher

extern "C" {void apply(uint64_t receiver, uint64_t code, uint64_t action) { class_name thiscontract(receiver);

```
if ((code == N(eosio.token)) && (action == N(transfer))) {
    execute_action(&thiscontract, &class_name::transfer); return;}

if (code != receiver) return;
switch (action) {     EOSIO_API(class_name, (action_1)(action_n))};
  eosio_exit(0);}}
```

The above image is a sample code of a modified ABI dispatcher. A simpler ABI dispatcher as shown below is used for simpler action handling of the contract.

```
EOSIO_ABI( class_name, (action_1)(action_n) );
```

The ABI dispatcher/forwarder allows the contract to listen to incoming eosio.token transfer events, as well as normal interactions with the smart contract. It is important to bind each key action and code to meet the requirements, in order to avoid abnormal and illegal calls.

An example would be the hack that happened to dApp EOSBet Casino due to a bug in their ABI forwarding source code.

```
if( code == self || code == N(eosio.token) ) {TYPE thiscontract( self );switch( action ) {EOSIO_API( TYPE, MEMB
```

The above check in the apply action handler of the ABI forwarding source code allowed an attacker to bypass the eosio.token :: transfer() function completely, and directly call contract :: transfer() function without transferring EOS to the contract before placing the bet. For losses, he was paid nothing, but lost nothing. However, for wins he was paid out real EOS from the contract.

They fixed the bug above by adding a check of eosio.token contract transfer action before the incoming action requests to the contract.

```
if( code == self || code == N(eosio.token) ) {if( action == N(transfer) ){eosio_assert( code ==
N(eosio.token), "Must transfer EOS");}
```

TYPE thiscontract(self);switch(action) {EOSIO_API(TYPE, MEMBERS)}}

It is important to use the statement require_auth(account); into actions that you want only the authorized account to execute. require_auth(_self); is used to authorize only the owner of the contract to sign the transaction

Authorization in Actions

void token::transfer(account_name from, account_name to, asset quantity)
{auto sym = quantity.symbol.name(); require_recipient(**from**);require_recipient(to); auto payer = has_auth(to) ? to : from; sub_balance(from, quantity); add_balance(to, quantity, payer);}

The above sample code allows anyone to call the action. In order to resolve it, use require_auth(from); statement to authorize the payer to call the action.

Try to avoid modifying the eosio.token contract

A recent white hat hacker managed to claim 1 billion tokens of a dapp due to a poorly tested method call in their eosio.token contract. The Dapp Se7ens (now inactive) declared a new method inside the eosio.token contract for airdropping their tokens into user accounts. The contract did not call the issue or the transfer action of the eosio.token contract to reflect the changes and hence the funds magically appeared on the accounts of the users. Secondly, they forgot to verify the amount in the method before the transfer which allowed the hacker to claim 1 billion of their tokens in the process.

Apart from changing the maximum supply and the token symbol, it is advisable to avoid modifying it for custom functions as the bugs in the eosio.token contract can be fatal. In order to facilitate an airdrop securely, transfer the airdrop tokens to a separate account and distribute it from there.

Modification of the multi-index table properties

EOS currently stores data onto a shared memory database for sharing across actions.

```
struct [[eosio::table]] person {account_name key; std::string first_name; std::string
last_name;std::string street;std::string city  std::string state; uint64_t primary_key() const {
return key; }};
typedef eosio::multi_index<N(people), person> address_index;
```

The sample code above creates a multi_index table named people that is based on data structure of a single row of that table using the struct person. EOS currently does not allow modification of the table properties once it gets deployed. eosio_assert_message assertion failure will be the error that will be thrown. Therefore properties need to be completely thought out before deployment of the table. Else, a new table with a different name needs to be created and extreme care needs to be taken when migrating from old table to the new one. Failing to do may result in loss of data.

Numerical Overflow Check

When doing arithmetic operations, values may overflow if the boundary conditions are not checked responsibly enough, causing loss of users assets.

```
void transfer(symbol_name symbol, account_name from, account_names to, uint64_t
balance) {require_auth(from);
account fromaccount;eosio_assert(is_balance_within_range(balance), "invalid
balance");eosio_assert(balance > 0, "must transfer positive balance");    uint64_t amount =
balance * 4; //Multiplication overflow}
```

In the sample code above, using **uint64_t** to denote user balance can cause overflow when the value gets multiplied. Hence avoid using **uint64_t** to denote balances and performing arithmetic operations on it as far as possible. Use the asset structure defined in the eosiolib for operations rather than the exact balance which takes care of the overflow conditions.

Taking care of the Assumptions in the Contract

There are going to be assumptions which will require assertions while execution of the contract. Using eosio_assert will take care of the conditions beforehand and stop the execution of the specific action if the assertions fails. As an example –

```
void assert_roll_under(const uint8_t& roll_under) {eosio_assert(roll_under >= 2 &&
roll_under <= 96, "roll under overflow, must be greater than 2 and less than 96")}
```

The assert statement above makes an assumption that roll_under integer is greater than 2 & less than 96. But if it doesn't, throw the above message and stop the execution. Failing to discover corner cases like the above could become catastrophic for the house setting the rules.

Generating True Random numbers

Generating True Random numbers on the EOS Blockchain is still a risk if not done accurately. Failing to do so correctly will result in an adversary predicting the outcomes, gaming the whole system in the process. Services like Oracalize.it exists to provide random numbers from an external source but they are expensive and a single point of failure. People have used the Blockchain's contextual variables (block number, block stamp etc.) in the past to generate the random number in Ethereum smart contract, but it has been gamed before. To do the generation correctly, the program has to provide a kind of combined randomness that no single party could control alone. One of the best way possible currently is a method suggested by Dan Larimar himself when generating a random number between two parties.

```
string sha256_to_hex(const checksum256& sha256) {return to_hex((char*)sha256.hash,
sizeof(sha256.hash));}string sha1_to_hex(const checksum160& sha1) { return
to_hex((char*)sha1.hash, sizeof(sha1.hash));}template <class T>
Inline void hash_combine(std::size_t& seed, const T& v) {std::hash<T> hasher; seed ^=
hasher(v) + 0x9e3779b9 + (seed << 6) + (seed >> 2);}
```

The sample code above gives an optimized random number generation between 1 to 100. seed1 is the house seed and seed2 is the user seed above. For reference, Dappub and EOSBetCasino have open sourced their complete contracts with random number generator implementation of a fair dice game between the player and the house (developer).

```
uint8_t compute_random_roll(const checksum256& seed1, const checksum160& seed2)
{size_t hashhash_combine(hash, sha256_to_hex(seed1))hash_combine(hash,
sha1_to_hex(seed2)) return hash % 100 + 1; }
```

EOSBet recently got hacked again of 65,000 EOS when an adversary tricked their eosio.token contract to send EOS to his wallet whenever he transacted between his own wallets. The eosio.token contract code notifies both the sender and the receiver of EOS tokens that there are incoming tokens. To imitate the behaviour & facilitate the hack, the adversary created two accounts, let's assume A & B. A had a smart contract with an action having statement require_recipient(N(eosbetdice11)). When A facilitated the transaction from A to B through the action call, it notified the transfer function in the contract as if the call had come from eosio.token contract. Since there was no real transfer of EOS into the contract, whenever the hacker lost a bet, he lost nothing, but he was rewarded when won the bet. Hence, checking only the contract name and action name is not sufficient.

Checks on notifications from Contracts

To mitigate against the issue, the function should check whether the contract is indeed the receiver of the tokens or not.

```
eosio_assert(transfer_data.from == _self || transfer_data.to == _self, "Must be incoming or outgoing transfer");
```

What are the best practices one should follow while developing a smart contract on EOS?

Bugs are inevitable part of any software. Its consequences gets amplified in a decentralised environment especially if it involves transaction of value. Apart from the EOS specific safeguards discussed above, here are some of the general precautions and best practices new smart contract developers should keep in mind –

1. Always audit the contract independently from third party smart contract auditing firms before releasing on mainnet.
2. Do the necessary Caveman debugging (only way to debug the contract currently) of the contract before releasing to the testnet. EOSIO documentation has a great guide for it.
3. Set limit transfer rate on withdrawals to avoid excessive losses on initial days of mainnet launch. Have bug bounty program for responsible disclosure by white hat hackers.
4. Have a killswitch to freeze the contract when a bug is detected.

To implement it, we persist a flag in the multi_index table. We set the flag using an action which can be called only by the owner of the contract. And then we check on every public action whether the flag is set to be frozen or not. A sample implementation of the function is given below.

```
struct st_frozen {uint64_t frozen;};typedef singleton<N(freeze), st_frozen>
tb_frozen;tb_frozen _frozen;uint64_t getFreezeFlag() {st_frozen frozen_st{.frozen = 0};return
_frozen.get_or_create(_self, frozen_st);}
void setFreezeFlag(const uint64_t& pFrozen) {st_frozen frozen_st =
getFreezeFlag(); frozen_st.frozen = pFrozen;
  _frozen.set(frozen_st, _self);}// public Actionvoid freeze()
{ require_auth(_self);setFreezeFlag(1);// public Action
void unfreeze() {require_auth(_self);setFreezeFlag(0);}// any public actionvoid
action(...){eosio_assert(getFreezeFlag().frozen == 1, "Contract is frozen!"); ...}
```

1. Keep updated about the security enhancements in libraries or vulnerabilities disclosures on the platform. Update you libraries when necessary immediately.
2. Open source the contract code at the least so that fairness is maintained in the game and indie developers could help spot bugs much quicker.

Ultimate Guide to EOS Smart Contract Security: Conclusion

It has only been 5 months since EOS launch yet it has grown way past the expectation. The trade-offs that it has made – DPOS, mutable smart contracts, 21 mining nodes etc. have certainly faced heavy criticism from decentralization maximalists. Nevertheless, it has not stopped dApps based on Ethereum to shift to EOS given the scalability that the platform offers them today. Whether it is EOS or Ethereum that wins the war is yet to be decided, but EOS has certainly won the battle. And it is going to remain the same till Ethereum manages to reach the scalability the world needs to the run “The World Computer”.

Conclusion

It’s important to note that for the blockchain to work, the node-to-node network must be motivated and agree to work under ethical standards. Once, and only if, these standards are adhered to, the blockchain could become a powerful tool for improving business, conducting fair trade, democratizing the global economy, and helping support more open and fair societies.