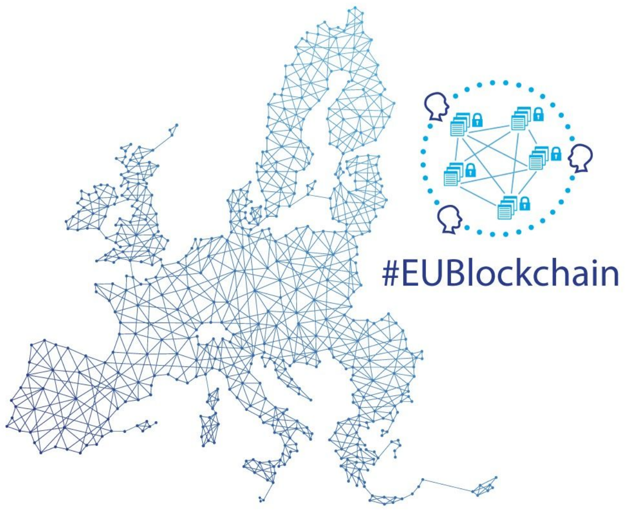
EU BLOCKCHAIN OBSERVATORY & FORUM

Workshop Report -
Cyber security –
Brussels, 29 October, 2019



Author: Tom Lyons
Published on 3 April, 2020
Comments and inquiries may be addressed to the following email: info@eublockchainforum.eu

Table of Contents

Presentation – Blockchain Security: The Difficulty of Protecting Digital Assets

Stefan Beyer, Cryptonics

- Blockchain technology is generally secure. The math is sound and the cryptography works.
- Digital assets held on blockchains are usually not secure. And this is a problem because these assets usually represent value. According to one report, USD 1.7 billion were lost or stole on blockchains in 2018. According to two reports in 2019, two criminal groups have been identified as being responsible for most incidents. This indicates that this kind of theft has become a business model, people are specialised on this.
- There are different areas of security concern on blockchains
    - Smart contracts, which are the only concern that is purely blockchain related
    - Off-chain code like databases and APIs
    - Infrastructure
    - Humans, which are the most serious security concern
- Indeed it can be said that humans interacting with offchain code on traditional infrastructure is the biggest problem

- Smart contracts make an interesting target because of their many vulnerabilities: they are publicly accessible, they hold value, and they are immutable, meaning bugs/vulnerabilities once the contract is released cannot be fixed.
- Smart contract security incidents are not uncommon. Famous examples include the DAO hack in which USD 70 million was lost, the Parity Multisig bug which resulted in USD 300 million being lost forever, the bug which would have made smart contracts insecure that was the cause of the postponement of the Ethereum Constantinople release. So one vulnerability is the underlying protocol changing.
- Beyer has seen this in his own work: In 22 smart contract audits he personally carried out in one year he found:
    - 16 critical issues, meaning exploitable vulnerabilities that would lead to loss of funds
    - 16 major issues, that may be exploitable
    - 36 minor issues that were not best practice
    - 0 flawless smart contracts, indicating how hard it is to get them right the first time.
- There are also the usual security issues off chain that people tend to overlook. These include security issues in:
    - Databases
    - Websites
    - Key management
    - APIs
        - This make ups of 80-90% of blockchain applications, and the security of which is often overlooked
- Common security incidents include
    - Unsecure authentication
    - Erroneous use of cryptography
    - Key storage exposure
    - Software vulnerabilities
    - Backdoors, commonly created by an unhappy employee
- Infrastructure vulnerabilities,including in node software or on the network
    - Node software, operating systems with vulnerabilities
    - Network
    - APIs that are open and shouldn't be
    - Permissions not set correctly
- Common incidents include
    - Vulnerability in the node software
    - OS-level Host Security
    - Network security
    - Denial of Service attack
    - Eclipse attack (isolate a node so that it only sees its own copy of the blockchain)

    - Exposed API
- Humans are generally the weakest link in security. Vulnerabilities include
    - Laziness, for example use of easy to crack passwords
    - Irrational behavior
    - Nondeterministic behavior
    - Valuing user experience over security, which admittedly is a very hard tradeoff in blockchain
    - Weakest link in the chain
- Common exploits of humans include:
    - Social engineering like phishing, which is a major problem
    - Information leaks
    - Insider attacks
    - Security shortcuts
- Blockchain protocols are secure, but there are possible attack scenarios. These include:
    - 51% attack
        - Chain reorganisation
        - Hashpower can be rented hourly
    - Selfish mining
        - when a miner increases his chance of getting a reward by pre-mining a block
    - Long range attack in proof of stake consensus
    - Poor randomness in proof of stake consensus
    - Delegator collusion in delegated proof of stake
- Case studies: Beyer then presented 3 case studies of security issues in blockchains.
    - Case Study 1 – Reentrancy, the DAO: Reentrancy is an exploit that was exploited in the DAO hack which resulted in a child DAO that was locked for 28 days. This led to the hard fork, which broke the law of immutability on the Ethereum blockchain. Interestingly, there are still reentrancy incidents 4 years later. Over those 4 years the best practice guidelines have changed several times, indicating that it is still not 100% clear how to deal with this problem. These changes are often driven by changes in the underlying protocol, an example of the protocol update dilemma. A big conflict between protocol updates, immutability and security. Smart contracts are expected to last a long time, to always work, but updates to the protocol can break them in unforeseen ways.
    - Case Study 2 – Front running: This exploits the fact that all transactions are public in the memory pool before they are confirmed. In this scenario the attacker overtakes a transaction, perhaps adding more gas fees to it, and gets its preferred transaction in first. By knowing what transactions are coming before they are confirmed, this also opens the door to a kind of insider trading on blockchains.

    - Case Study 3 – Protocol attacks: We have seen a large number of attacks on different protocols. There have been 51% attacks on Ethereum Classic, Vertcoin and Bitcoin Gold. There was an attack on the Verge blockchain exploiting timestamp manipulation. The Lisk blockchain suffered an unclaimed address attack.
- There are many ways to classify security tradeoffs, and designers of blockchains need to take these into account. Among them:
    - Usability vs Security issues
        - Usability = Efficient, Easy
        - Secure = Difficult to use
    - CAP Theorem, which has been known for a long time, states that in a distributed data store it is impossible to simultaneously have more than two of the three properties: consistency, availability, partition tolerance.
    - Blockchain Trilemma, which states that in a blockchain you can only have two of the three properties: security, decentralisation, scalability

Presentation – How to avoid vulnerabilities in smart contracts

Joran Honig (MythX)

- When looking at smart contract security, you need to consider the following smart contract properties: immutability, censorship resistance, high security requirements, high-target code as it often contains value
- Smart contracts have some desirable security qualities, like immutability and censorship resistance.
- However, since they tend to hold value, they are also targets for hackers. So they have high security requirements too
- Smart contracts are code written by humans, so subject to bugs and other vulnerabilities.
- Example of a smart contract could be someone with a couch to sell to the first person who sends the owner €300.00. Such a simple use case contains tricky edge cases. For example, if a potential customer sends €300.00 over the blockchain, the smart contract owner could try and quickly send him or herself €300.00 as well. If this arrives first, the conditions of the smart contract would be fulfilled, the potential customer's money would arrive, and the owner would have the money and the couch. This is a simple illustration of front running, and serves to show that even simple contract situations can have complex ramifications that need to be taken into consideration.
- Examples of smart contract incidents include:

    - Parity incident - accidental: a user accidentally committed an action that self-destructed a smart contract that was part of a wallet that effectively froze $300 million in assets forever.
    - Traditional cyber security issues: The speaker gave an example of the discovery of a batch overflow bug that could effect multiple ERC20 smart contracts. This was published before any exploit could be carried out, but illustrates the existence of traditional cyber security vulnerabilities in smart contracts.
    - DAO attack: this too was a vulnerability that caused a potential loss of funds. To fix it, the transaction history had to in effect be rolled back by the community.
- The question is what can we do to prevent security exploits like the DAO hack or Parity bug? Options include:
    - Conduct a full, manual security audit. There are no doubts about the merits of this as it is thorough, but it is also expensive and generally is done one time, usually at the end of the process once the contract has already been written. Due to expense, this step is also often unfortunately skipped. Also, it can be more expensive to fix problems after the contract is finished than it would be to identify and deal with them during the development process.
    - Use program analysis. Considering the above, it is better if possible to do the security analysis during the development phase. This only costs computational resources, so is cheaper than a post facto audit.
- Techniques for program analysis include dataflow analysis, symbolic execution, deductive verification among others, techniques that try to explore all the different interpretations of the code and flow through the contracts to check for patterns. This can reveal some vulnerabilities.
- Some program analysis techniques require setup on the user end. Others can be fully automated. The latter is known as Automatic Analysis.
- MythX focuses on automatic analysis. Specifically, they employ symbolic execution, grey-box fuzzing and abstract interpretation and dataflow analysis. These techniques take a smart contract and try to simulate the different interpretations and ways a contract can be executed, so follow the flows through the contract. This allows researchers to identify certain patterns and rules, which in turn can indicate vulnerabilities.
- By identifying vulnerabilities researchers can also identify various vulnerability classes, make models of them, and then create a set of logical rules that precisely describe the weaknesses they represent, and which subsequent analytical tools should look for.
- Use cases for Automatic Analysis include:
    - Quick iteration during development, giving developers quick iterative feedback as they work.
    - Thorough code reviews during testing.
    - Risk Assessment, providing users an easily understandable description of the issues that were found.

- Such techniques can also be used by auditors to extend their capability by creating formal models of the business logic of the contract they are auditing, and using this for analysis with the tool.
- The techniques described allow auditors to build in security from the start and create public visibility into the risks of a given smart contract.
- QnA:
    - What is the current state of play between what developers can do with MythX and what they need.
        - Main different between security analysis and audit, in security analysis the MythX team is forming the models of things to look for.
        - However, if the vulnerability is not generic, it can slip through such an analysis.
        - So today one main value of a tool like MythX is to extend the capability of auditors by allowing them to automatically find generic, well-known and/or previously modelled vulnerabilities.
        -

Presentation – Security aspects and open challenges of blockchains and consensus protocols

Stefano De Angelis (University of Southampton)

- Blockchain is a Multi-Party Infrastructure based on a distributed network of actors working on a common task who may trust each other and need to communicate and transact over a shared infrastructure. There are a wide range of use cases for this. Transactions could be purely economic value, as is the case with cryptocurrencies and general financial services use cases involving blockchain-based digital assets, it could be value and information, as is the case in supply chain, or it could automated machine-to-machine transactions of data and/or value in IoT. In all such cases the data must be secure.
- In the past we used centralised solutions that entrust a third party to regulate the interaction between the parties, control the correct behaviour, manage the transactions. That raised the question of whether you can trust the third party, and also made the third party a single point of failure.
- Blockchain deals with this by distributing trust among all the participants in the network using cryptography and various means of validation. The blockchain itself seems secure, and so for many it is the most promising technology at the moment for distributed trust, it is worth looking at blockchain performance and security in detail before to be sure if this is so.
- To do such an analysis you need to understand the different types of blockchain.

  - public blockchain, where any node can join the network and access/write data in the ledger, for example Bitcoin or public Ethereum
  - private blockchain, in which read/write operations are restricted to authorised participants, such as is the case with Hyperledger Fabric.
- You also need to differentiate between permissionless and permissioned blockchains. In permissionless blockchains any node on the network can read and write transactions. In permissioned blockchains, one or both of these actions is restricted to authorised nodes.
- Multi-layer architecture
  - application layer
  - smart contract engine layer
  - consensus layer
  - network layer
    - the last two are important for security, as the top two layers are dependent upon them.
- As stated by Cachin & Vukovic staed, blockchains can and do make use of various techniques that are available for reaching consensus, replicating state (i.e., agreeing on a single version of the truth across the network), and broadcasting transactions that help them deal with the security and other challenges of operating in environments where network connectivity is uncertain, where nodes on the network can become subverted by an adversary, and where interactions among nodes are inherently asynchronous.
- In a potentially insecure or hostile environment like the Internet, consensus protocols allow parties to reach agreement on what transactions to put in the ledger and in what order.
- To be successful, a distributed consensus protocol must guarantee safety (integrity of the data) and liveness (high availability of the network). This must be done in the face of different network models, different trust assumptions among participants, and in light of the possible existence of malicious nodes (referred to as byzantine nodes).
- The problem of how to coordinate nodes in a distributed network that relies on consensus but in which some nodes may be malicious is a well-known problem in computer science often referred to as the Byzantine Generals Problem, and has been studied for more than 20 years. The advent of blockchain has renewed interest in this problem. Blockchain protocols generally try to be Byzantine Fault Tolerant.
- There are two major families of consensus protocols in blockchain:
  - Lottery based: Probabilistic election of leaders in the network; good for public blockchains.
  - Voting based: the classic BFT protocols, they use a voting process to elect a leader. They require an intensive message exchange, and so are good for smaller, permissioned networks.. Practical Byzantine Fault Tolerance is the most famous implementation of BFT.
- **Proof of Work** is probabilistic, miners vie to win a lottery and whoever wins can propose a block. This is good for the public because it supports a large number of nodes.

Someone will always win the lottery and the others can then validate. This is however expensive in terms of latency and energy used. It does guarantee a high degree of liveness, as the network is highly replicated. Safety however is only guaranteed eventually. That is because on PoW on a large network, two miners could be chosen to lead simultaneously, leading to a fork in the chain. This only resolved over time as the network will converge on the longest chain. So finality requires time.

- **PBFT.** Replicas on the network propose a leader, when a leader has a block, nodes exchange a set of messages to agree on the order of the message. Once the messages are received they commit immediately. So finality is immediate. This has been proven to work if 2/3 of the nodes are honest. Liveness is guaranteed as long as the network is sufficiently synchronous. In a large network with many nodes, performance degrades.
- Comparing the two approaches, we see that lottery-based consensus protocols offer eventual consistency (finality is available after a certain amount of time), high availability (large networks are robust), but slow performance (the process takes a long time). Voting-based consensus approaches offer high throughput (performance is fast), strong consistency (finality is immediate upon the vote), low availability (as the network is smaller and so more vulnerable to attack or failure than a large PoW network), and bad scalability (performance degrades as the network gets larger).
- If PoW and PBFT represent two ends of the scale, there have recently been many new approaches and algorithms proposed that attempt to combine the best of both worlds: to achieve good performance, good security and good scalability. Many claim to provide this while tolerating more failed nodes than in classical PBFT. Southampton calls these hybrid consensus algorithms.
- An important new family of consensus algorithms is **Proof of Authority (PoA)**, the goal of which is achieving eventual consensus and high performance, while allowing for reasonable scalability. It combines PoW and PBFT, and only works in permissioned networks in which the nodes are known. They are generally based on a mining rotation protocol in which a leader is elected to propose the next block.
- The presenter described in detail two PoA proposals for Ethereum: **Aura** for the Parity client, and **Clique** for GETH.
    - In the **Aura PoA** protocol all the authorities (validators) are assumed to be synchronised, and time is divided into steps of a fixed length. At each step, a new leader is elected in a deterministic way. The leader proposes a block and broadcasts to the replicas. The block is not committed immediately, however, because it is possible that a leader misbehaves by a) not proposing a block, b) proposing more than one block, c) proposing different blocks to different authorities. The majority of authorities can vote this leader out. During this process the chain can fork. Finality (the committing of a block), therefore, only occurs after a certain time period, namely once a majority of authorities have proposed their blocks and they have been accepted.
    - The **Clique PoA** protocol also proceeds in steps, but they are not reliant on time. A leader is determined mathematically but there is also a subset of other possible

leaders that can propose new blocks. There are forks but this only happens if the leader and others propose blocks simultaneously. It is resolved by a scoring mechanism based on Ehtereum's GHOST protocol. Finality is therefore also achieved only over time.

- Clique and Aura are more performant because of low number of messages. Increasing nodes affects performance but they are more scalable. The researchers however have found some security issues, with some potential synchronisation problems in Aura and consistency problems in Clique.
- Considering the amount of research and development going into consensus algorithms today, and the number of new technologies being proposed, it is important to define a general framework for benchmarking permissioned blockchain platforms. Such a benchmark would help illuminate tradeoffs between performance and security, among other things.
- During the QnA one participant noted that certain projects, like Cardano and Ethereum 2.0, which in his opinion are taking the time to do things properly, get criticised for being slow to release. He also pointed out that there seems to be a big gap between academic research and what companies are doing in practice, where they do not necessarily take time to do rigorous mathematical analyses of their protocols.
- Another asked about how to organise protocol benchmarking. This would be useful for example in the context of EBSI or generally at the EU level as a service for the community and to improve the general security of blockchains. Also, it was asked whether such benchmarking should be confined to the infrastructure layer or should be employed at the application layer too.

Presentation – Zero Knowledge Proofs and Blockchains: what usages beyond confidentiality

Nicolas Liochon (Pegasys)

- The point of a zero-knowledge proof is to be able to prove mathematically that you know something without actually revealing what you know. The idea is quite old, it was invented by Micali and Goldwasser at MIT, and resulted in a Turing prize for them in 2012. While the original work was done decades ago it has only been recently, with blockchain, that significant uses have been found for it.
- The concepts and the technology are very complex and difficult to understand for non-specialists. To illustrate, Liochon used the example of a Finding Waldo illustration, where you could prove that Waldo existed and that you knew his location but not reveal anything else in the illustration by blacking out all of the illustration except Waldo.

- Another illustration provided was proving that you had the correct answer to a Sudoku puzzle without revealing that answer. In this case Proof that you know something without revealing what you know. In this case, you have public information (the unsolved puzzle), private information (your correct solution), and then a number of statements which represent a set of conditions that, if proven, would also prove that the statement is true.
- More formally, ZK proofs work in this way. There is a public input, a private input, and a number of statements, referred to as a circuit. Then there is a prover, a program that proves the statements in the circuit based on the public and private inputs, and a verifier, which is something another party can run to verify that the proof is true without discovering what the private input was.
- One advantage of ZK proof is that doing the work of verification is far less computationally expensive than doing the work of the actual proof. That makes it scalable: the proof is done once but can be "proven" to others many many times without redoing that work.
- ZK proofs are also non-interactive, meaning that the prover can release the proof but does not need to be present for the verifier to verify it.
- ZK proofs can be used for creating private transactions on a blockchain, as is done for example with ZCash or Aztec.
- In Aztec "notes", which are encrypted representations of some abstract value, like the details of a transaction, are kept in a registry. Notes can be exchanged, but without revealing the amounts associated with them (the content of the note). A transaction involves the destruction of some notes, the creation of new notes and, crucially, the ZK proof that you are not creating any new money (double spend). Important is that in this part, the parties to the transaction are not hidden, simply the amounts. Providing confidentiality to the parties involved is handled in a different, non ZK, part of the protocol.
- An unexpected but interesting use case is cryptographic sortition, using ZK proofs to select a subset of participants from a larger group such that only the participants know they have been selected, and can choose when they want to reveal themselves with this proof that they have been selected. This can be helpful in cyber security in cases where you may need to select participants, for instance to do some work, like validators on a network. If these are known ahead of time they can be attacked. With cryptographic sortition, such participants can do their work in secret and only reveal themselves, and the proof that they had been selected, after the work is done. ZK can help make cryptographic sortition practical.
- A very exciting use case for ZK proofs outside of security is in the realm of scalability. As many know, the blockchain mantra is "dont trust, verify", and to maintain the integrity of the chain, all the nodes always check the validity of all transactions, and all (full) nodes need to store a full copy of the chain (the "state"). The more users and more transactions, the larger this becomes.

- With ZK proofs it becomes possible to verify a proof without executing it. In this way, one node could verify the transaction and provide a proof, and all the other nodes would just need to verify the proof, which costs much less in terms of computing power and storage. This could greatly increase blockchain capacity.
- Such a solution is known as a rollup. Rollups are a Layer 2 solution, so not part of the blockchain protocol but something that runs on top of it. In a rollup a user sends a transaction not to the blockchain, but to an operator that validates the transaction. The operator can then send the proof that the job was done correctly to the blockchain.
- There is a lot of effort going into ZKP, and new zero knowledge schemes now appear almost weekly. All have different properties in terms of the time needed to generate the proof, the size of the proof, time needed to check it, whether or not it is quantum resistant, and how the process to initially set up its initial randomness (generally known as a "ceremony").
- ZK proofs can benefit from improved hardware, including chips with more cores, using graphics processing units (GPUs), or more sophisticated hardware like field-programmable gate arrays (FPGAs), or application-specific intergrated circuits (ASICs).
- We can expect initial rollout of rollups soon, within a year, but they won't provide full benefits in terms of increase in throughput. It will likely be 5-10 years for a full mathematical solution that is proven to be the optimal solution.

Working Session – Leveraging blockchain for the core cybersecurity aspects.

- The working session which concluded the day was dedicated to use cases in which blockchains might be employed to increase cybersecurity.
- Cyber security is often described as a question of preserving the following three properties (often referred to as CIA)
  - **Confidentiality:** protecting data against unintentional, unlawful or unauthorised access, disclosure or theft. Blockchains could be used to support confidentiality through managing access rights or registries, for instance in self-sovereign identity.
  - **Integrity:** maintenance and assurance of the accuracy and consistency of data over its entire lifecycle. Blockcchains can support integrity through things like time stamping and notarisation of date, for example in document certification.
  - **Authenticity:** preserving the genuineness of data and making sure that data received at the collection server is original and was received exactly as sent. Blockchains can support authenticity through providing digital credentials and ensuring digital scarcity, for example in track and trace and payments systems.

- Specific use cases include food safety, tracking blood diamonds, pharma safety, medicines, property title for immovable assets.
- Confidentiality is an important element in blockchain IoT use cases.
- Confidentiality is an important element to support blockchain consortia. It can help make it easier for competitors to work together on common platforms and also share sensitive services, like KYC.
- Blockchain, one participant said, was not really made for confidentiality. Bitcoin does integrity and authenticity well, but not confidentiality.
- Blockchain can be used for e-voting systems, as there are hard requirements for confidentiality but also integrity. This has been done in Slovenia, Switzerland and for referenda in Brazil.
- Other use cases include: asset tokenisation, custody and escrow of digital assets, provenance and tracking, accounting and reconciliation, digital identity, real-time transactions, micro-payments and funding, automated execution.
- In pharmaceuticals, blockchain can help with efficiency, as the science changes quite rapidly, but it takes a long time for new approaches or recommendations to become widely known and accepted. A blockchain-based public consortium between family doctors, labs, hospitals, etc., using authentic, confidential and integral data, could make it easier to securely adjust recommendations in terms of treatment, as well as allow AIs to work with large data sets while preserving privacy.
- One problem with medical records is that there often are no access logs: so people don't know who has accessed their information. Perhaps blockchain could help with such systems, as is being done now in Estonia.
- Blockchain could also be used for track and trace on information, and so help combat fake news by providing easily available provenance information. This is already happening with projects to provide consensus for oracles (data sources for blockchain, for example stock price or weather data). This principle could be extrapolated to news.
- Another use case that is becoming popular is providing unique, scarce digital assets in video games.
- EUIPO, the EU Intellectual Property Office, has been working with blockchain to fight counterfeiting, tax evasion, or ensuring authenticity of software. Yet these things can also be done without blockchain too. But, as one possibility, you could manage certificate authorities in a decentralised way on a blockchain and perhaps a reputation system for authorities.
- Autonomous cars are a potential use case as well, as each car needs to be identified and to be able to identify other cars. There are GDPR issues here among others. Perhaps blockchain can help add confidentiality to the data.
- In security use cases, blockchain is probably most useful in situations where decentralisation helps security, for example managing domain names (DNS). Self sovereign identity solutions – where an individual or entity collects provable attestations and uses these for identification, authentication or providing assurances to others, as

opposed to relying on a single third party to authenticate – are also good blockchain use cases.

- There was a discussion about secure authentication, information sharing and access right management in identity systems. The question was whether it is more secure to store credentials/identities etc. in a decentralised manner on a blockchain to combat phishing and similar attacks, and also to allow individuals to grant and revoke permission. It was pointed out that there are already good options, like PGP, but they suffer from usability problems.
- This was followed by a discussion of using blockchain to improve service resilience and availability. Blockchains are distributed, which makes them highly resilient to attacks like DDOS. This was one of the oldest use cases. It was questioned however if people were really using blockchain for this. One area where it can definitely be used is in decentralised naming registries, DNS. You can go a long way to making it very hard to hack a DNS and take over an identity.
- Blockchains are secure, so attackers will go somewhere else, for instance to the point of interaction with the blockchain. The weakest link.
- You also need decentralised hardware to make this work, like Swarm or IPFS. Because as useful as the blockchain is, if attackers get to the hardware, etc., then it doesn't matter. Most blockchains would go down if Infura went down. It is one of the main gateways.
- The discussion ended by looking at key priorities and recommendations in the following areas: a) what are companies most worried about in terms of cyber security, b) what are the most important research priorities, c) what are the most important use cases to be prioritised to radically increase resistance against attacks, and d) what changes/additions to regulation could help?
- One problem is that not enough people understand cryptography and how to use it properly. Lots of blockchain projects are not using it properly or developing their own cryptography, and it doesn't necessarily work. So we need more education and expertise.
- Smart contracts need research into how to make them more secure. Lotsa people researching consensus algos, less into secure smart contracts. This too is an issue of education. Many problems are not new here: the reentrancy problem is well known.
- One way to make smart contracts more secure is to develop smart contract languages whose logic is limited to the most important use cases. So something between the Bitcoin script language and the full Turing-completeness of the EVM. Many people use smart contracts to implement complex things that are unnecessary, and can be done better with conventional means. Just use smart contracts for things they are really useful for. If you had a robust enough but reduced set of logical building blocks covering most smart contract cases, it would be much easier to use and debug. Full Turing completeness makes for vulnerabilities. The problem there is what happens whe you do have a legitimate use case for which these building blocks don't apply.

- One big worry but also a use case is track and trace in the development of complex systems, like software for airplanes. Here we need better audit trails about who did what in building components, so you can identify faults hopefully ahead of time and save lives, or be able to identify mistakes after the fact and assign liability.
- Blockchain can help prevent collusion or adversarial actions among partners in a consortium
- The working session participants ended with the following recommendations:
    - Incentivise responsible vulnerability disclosure. The incentives in BC are different, because economic incentives are so strong. So a large incentive not to acknowledge problems that could affect the economics of a bc network.
    - Developers / actors in the blockchain space should leverage the tools that are out there that let you formally verify or make informed security decisions based on mathematical facts you can discover from these protocols and contracts.
    - This is important because you have real assets on blockchain, so audits are real important. So the question is should it be a regulatory requirement? The problem is that this could be a barrier. So maybe education is more important than more rules. Make it clear to people how major the consequences can be, especially in light of immutable contracts.
    - Standardisation is one thing, regulation is another. Another way to deal with this is certification, which is different from regulation. Some set of quality certificates for smart contracts etc. could be a good middle ground.
    - There is the question of the impact of quantum computing. People tend to think that quantum spells the end of blockchain. But if you talk to security experts, they say there is no problem there are already quantum resistant algorithms. So one thing the Observatory could do is do a better job educating people on that aspect.

Appendix

Workshop slides

- CyberSecurity Workshop - Workshop Slides
- Blockchain Security - Stefan Beyer - Cryptonics
- Security Analysis - Joran Honig - MythX
- Security aspects and open challenges of blockchains and consensus protocols - De Angelis - Southampton
- Zero Knowledge Proofs – Nicolas Liochon - Pegasys

Workshop videos

- Videos from this and all other workshops can be found on the EU Observatory website under reports.

- Videos specific to this workshop:
    - Cybersecurity workshop Part 1
    - Cybersecurity workshop Part 2
    - Cybersecurity workshop Part 3

Official agenda

Time	Activity
9:30	**Registration & Welcome Coffee**
10:00	**Introduction of the day - Agenda and objectives of the day**
10:15	**Presentation – Blockchain Security: The Difficulty of Protecting Digital Assets** Stefan Beyer (S2 Grupo)
11:00	**Presentation – How to avoid vulnerabilities in smart contracts** Joran Honig (MythX)
12:30	**Presentation – Security aspects and open challenges of blockchains and consensus protocols** Stefano De Angelis (University of Southampton)
12:30-13:45 Lunch break	
13:45	**Presentation – Zero Knowledge Proofs and Blockchains: what usages beyond confidentiality?** Nicolas Liochon (Pegasys)
14:30	**Working Session – Leveraging blockchain for the core cybersecurity aspects: Integrity, Confidentiality and Authenticity** Content: Interconnection and use cases between cybersecurity and blockchain (security for identity systems, authentication, traceability, payment systems, DNS…) What role for blockchain technology in the context of the Cybersecurity act?
16:00	**Conclusion**
16:15	**End of the day**