STRATUS CYBER

BLOCKCHAIN AND CRYPTOCURRENCY-ENABLED PRODUCTS AND SERVICES

KNOWN SECURITY CONCERNS

Introduction

Apart from the massive interest within the blockchain-startup community towards successful execution of ICOs, there has been considerable traction in the context of products and services built on and around the Ethereum and Bitcoin blockchains in particular, and several newer blockchain technologies in general.

By far, the most important instances of these products and services include wallet software, cryptocurrency exchanges, and ERC20 token trading platforms like EtherDelta. Based on the blockchain community's experience with popular products in this domain and related infamous security-breaches/thefts, we have begun to craft a structured approach towards addressing the most common security concerns; in addition to formulating guidelines that deal with a broader cybersecurity-perspective on employing blockchain technology securely.

Many of these guidelines, especially those that pertain to smart contract code and wallets, are generally useful in secure designs of ICOs as well.

Cryptocurrency Exchanges

To understand the multi-faceted security concerns that arise in software infrastructures built around blockchain technology, a comprehensive guide to the ecosystem of cryptocurrency exchanges is the most apt place to begin. Regardless of whether your product/service is an exchange or not, it is highly likely that it would employ building blocks, processes, and software used to run exchanges - after all, any product employing blockchain/cryptocurrencies needs to build the bridges that mediate between the off-chain and on-chain world, just like exchanges.

Ecosystem Fundamentals

Before delving into the security risks and details of known breaches, it is important to analyze the fundamental entities and building blocks involved in the functioning of an exchange. At its most basic level, an exchange ecosystem involves:

A Web Application

The core of an exchange, is the web application that powers it. Essentially, this is the user-friendly online GUI developed by the company setting up the exchange, and as such, it is centrally owned. This web application is the "middle-man" that arbitrates between end-users, and the public blockchains, to move digital assets.

Public Blockchains

The immutable truth of digital asset ownership, rests on public blockchains of technologies such as Ethereum, Bitcoin, Ripple, IOTA etc. As any movement of cryptocurrency has to be ultimately initiated and settled on the public chains, interaction with and monitoring of these chains is a vital part of the cryptocurrency exchange ecosystem.

Wallets

The term "wallet" is often used in a rather non-specific and ambiguous manner within the blockchain community. Often, it is aimed at the wider general audience, to signify software in charge of storing and spending cryptocurrency. For the technically inclined, this definition is vague, and this vagueness has serious security implications.

To understand the subtle nuances of cryptocurrency "wallets'; we must be clear about the following foundational concept: technologies such as Ethereum and Bitcoin, have defined reference"clients" (software that follows the fundamental protocols of the technologies to transact cryptocurrencies), on top of which, "custom" software is built by a large number of companies and exchanges - all of which loosely falls under the category of wallets. This includes software that users can download on personal PCs and devices, hosted software, and proprietary software developed by the exchange for its own interaction with the public blockchains. Furthermore, all transactions depend on the ownership of the private key (the string which gives complete control to a cryptocurrency account), and different software implementations of wallets, deal with private keys and their hosting/ownership in different ways (for example "hot" on line wallets store them on line, and "cold" offline wallets store them offline). Finally, In addition to these differences, we have the concept of the wallet as a smart contract on the Ethereum blockchain - a contract that holds Ether; a concept which further muddles up the notion of what a wallet is, unless properly specified.

To summarize, not all wallets are made equal, and the term "wallet" does not define a specific implementation.

Security Considerations

Each one of the three fundamental constituents of an exchange mentioned above poses its unique security challenges - challenges that have been highlighted through major breaches in recent times.

Central Web Application

Interestingly, many of the highly publicized "cryptocurrency hacks" could have been avoided by utilizing age-old wisdom which relates to good cybersecurity practices around Internet startups and web applications in general.

- **Protecting Credentials:** In August 2017, Enigma was hacked through simply acquiring and using the CEO's login credentials for the website. Once the hackers had access to the website, they could easily disperse false information and make investors direct funds to addresses of their choosing. $500,000 in Ether was stolen in this hack, which had nothing to do with the Ethereum blockchain itself. The CoinDash website suffered a similar fate, with hackers gaining access and changing the receiving contract's address. The 2011 Mt. Gox attack was made possible by gaining access to the auditor's credentials. All such instances reinforce the importance of a strict control of credentials to a company's digital presence, which is nothing new, but surely deserves a mention.

- **Version Control:** Although a rather obvious software development approach taken by developers of large software, using VCS (Version Control Software) was sidelined in the case of Mt. Gox - a decision which was later attributed to the general lack of security in the Mt. Gox enterprise. A VCS helps track changes made to software - when they were made, where they were made, and who made them. It also allows developers to roll-back to an earlier version of the software in case a recent change has unwanted implications. We believe that a VCS could have saved Mt.Gox from a number of security problems, and may even have averted the hacks

- **Software Testing:** Infamy surrounds Mt.Gox in this domain of best practices as well. It is common practice to have software testing policies in place for even non-system-critical code; and in the case of software handling large sums of money, testing (at the level of unit tests as well) becomes essential. Unfortunately, Mt.Gox did not have one in place.

- **Protecting IT Infrastructure:** The BitFinex hack is a prime example of failure to properly guard a company's most basic point of failure - its central servers. Hackers attacked the servers, got hold of the API which was being used to communicate with BitFinex's security solution "BitGo'; and made the necessary calls to transfer BTC from user accounts to addresses of their choosing. Again, the hack did not exploit a security loophole in the Bitcoin technology, rather the carelessness in the context of how the BitFinex central system was set up.

- **DDoS Protection:** Owing to the extreme mismanagement and general state of frenzy and exuberance within the blockchain community, many traditional security concerns are overlooked - and this includes the good-old Distributed Denial Of Service attack. With trade volume in the millions each day, cryptocurrency exchanges can not afford to have their services dropped due to DDoS attacks - the consequences are much worse given the volatility of market sentiment and the fragility of reputations in this nascent space. Towards this end, it is essential to have basic DDoS protection - the benefits far outweigh the cost.

- **Special Attention to Edge-Cases:** In March 2014, the Poloniex exchange lost 12.3% of its total bitcoin supply owing to code that was not designed to handle a special scenario: a large number of simultaneous withdrawal requests. In this delicate world of digital assets, it is essential to think of edge-cases as a make-or-break critical design element, rather than a run-of-the-mill server error.

- **Internal Application Monitoring:** As a final, and essential measure, internal monitoring of unusual activity by custom-built security software is often the most effective fail-safe. Even in the Poloniex hack mentioned above, a more massive loss was averted owing to the software that was in place to check for unusual withdrawal activity on accounts. Such a system was not present in the case of Mt.Gox, and the company had to find out about the “unusual activity” the hard way - when their customers sounded the alarm to delayed transaction times.

Blockchain Technology

As stated earlier, almost all popular cryptocurrency hacks (including the Ethereum DAO disaster which resulted in a split within Ethereum) have targeted vulnerabilities in the infrastructure built around or upon blockchain technologies - they have not been a product of weaknesses intrinsic to the underlying blockchain technologies. Regardless, there have been certain flaws in the technologies themselves which, although corrected (or are being addressed) by the core developers, deserve a mention as they are relevant to understanding the design of blockchain-enabled software, especially wallets (which we discuss in the next section).

- **Bitcoin Transaction Malleability:** This particular security flaw pertaining to the structure and processing of transactions in the Bitcoin blockchain has been addressed for the most part in BIP 66 (Bitcoin Improvement Proposal# 66). Nevertheless, as it has been exploited in the past and influenced wallet/exchange software design, it is vital to understand its mechanics. Transaction malleability, as the name implies, refers to changing transaction data. In the Bitcoin context, the following malleability possibility existed: the format of the digital signatures could be changed, while still being valid. This in turn, changed the hash of the entire transaction, which is used as the ID for the transaction. Thus, if the changed transaction is accepted first (because it is still a valid transaction with a valid signature), any entity monitoring the original transaction ID, would see its transaction as unconfirmed - opening up a range of malicious possibilities.

- **Ethereum Security Considerations**: Being much more complicated than the Bitcoin blockchain protocol, Ethereum lends itself to a greater range of possible malfunctions - problems that can have significant impact on the security of applications built using Ethereum. An exhaustive discussion on the Ethereum Virtual Machine (EVM) and its programming pitfalls is beyond the scope of this document, but summarizing the dangers, it is important to acknowledge shortcomings in the way code is structured and written for the EVM. This includes accounting for how EVM processes addresses, fall back function calls (calls to the unknown), type casts, gas less sends, reentrancy attacks, overflows/underflows for uint256 data, and other known areas of potential security threats. Complete and detailed testing and code auditing of EVM code by experts, is thus essential.

There are a number of new blockchain technology startups gaining traction as well, and while the ones that have the strongest community around them continue to show some cracks in security and scaling, the newer technologies claim to address these shortcomings.

Wallet Software

Keeping in mind the fact that the term "wallet" is an umbrella-term covering many types of implementations and contexts, the security considerations around wallets broadly fall into the following categories:

- **Ethereum Contract-Code Wallets:** The Ethereum public blockchain hosts smart-contracts - some of which are programmed to transact and hold Ether, in essence becoming wallets. All crowdfunding endeavors based on Ethereum, including the famous DAO initiative, thus have contract-code that is acting as a store of Ether. Such wallets/contract-code are a critical point of failure if improperly coded/managed. A simple recursive call/reentrancy attack that exploited a loophole in the code that dealt with reimbursement of Ether for returned ERC20 tokens, led to around $50 Million worth of Ether being stolen. More recently, carelessness on the part of Parity's multi-sig wallet design, resulted in $150 Million worth of Ether "accidently" being locked up. The reason? To simplify design and save gas costs, the user-end wallets developed by Parity (wallet software being run on user's computers to interact with the public blockchain) were instructed to reuse code already residing on the public blockchain - a smart-contract being used as a library. Unfortunately, due to improper security audit of its code, a user was able to take control of all of the wallets being managed by it (by taking ownership of this library contract itself), and accidentally killed the contract, freezing all the funds. In essence, "wallets" on the Ethereum public blockchain, or parts of them that reside on it, are subject to intricate attacks that we are gradually beginning to understand. Needless to say, security audits and extensive testing of these contracts is the way forward towards smooth, secure operations.

- **Multi-Sig Wallets:** Transactions on cryptocurrency blockchains require digital signatures produced with private-keys. While a single private key can signify ownership of an account, its usually much more secure to require more than one signature, corresponding to more than one private key. This is the functionality multi-sig wallets provide (possible with both Bitcoin and Ethereum), given that the account (public address) being managed is a multi-sig account on the relevant blockchain. The use of these wallets (and in general, the multi-sig "concept") is common in cryptocurrency exchanges as well - with the exchange holding only one of the private keys, the others being managed by the users. Hackers would need to, in essence, employ multiple hacks to be able to have the "n out of m" signatures required to initiate transactions. However, mismanagement and poor design has been a key indicator of security flaws within the multi-sig approach as well. An example being the Bitfinex hack in August of 2016. Bitfmex had employed the multi-sig hot wallet approach by using BitGo's services. Three signatures were required - two from Bitfinex itself, and one from BitGo, requiring hackers (in theory) to hack both Bitfinex AND BitGo to manifest a successful breach. In practice however, the design was flawed and the attack only required the hackers to take control of Bitfinex servers, and then use BitGo's APIs to instruct it to validate transactions as well - defeating the whole idea of using multi-sig wallets. Thus, a careful multi-faceted approach to code testing, software architecture, and product design in the context of security is required to build solutions that can stand their ground in this new, "wild" world of cryptocurrencies.

- **User-Wallets:** Wallets managed by users themselves come in a number of "flavours" - each exposing their own unique security concerns. Primarily, either the wallets are software wallets or hardware wallets. Hardware wallets, like the Ledger Nano S, are the most secure - the private keys reside offline, on the hardware device, and never leave the device. The signed transactions are transmitted from the device through the USB interface, and thus it can be used to sign-off on transactions without ever having to host the private key on a user's computer that is connected to the Internet. A DIY hardware wallet can also be implemented by simply using a signing-software where the private key resides on an offline computer, and then using a USB stick to transfer signed transactions from this offline computer to the one connected to the Internet, and use additional wallet software to broadcast signed transactions. DIY hardware wallets and wallets like the Ledger Nano S are more prone to user-error (forgetting the PIN to the device, losing the offline computer, losing the private key etc) than to hacks - which are virtually impossible if hardware wallets are used correctly. Software wallets, on the other hand, especially "hot" wallets which are connected to the Internet, host the private key on the computer and thus are much more vulnerable to attacks. Additionally, most of these software wallets rely on lightweight nodes of the respective public blockchains for their mediation with the blockchains. Lightweight nodes are subject to being fooled by malicious users, and this could be potentially catastrophic for wallet software that relies on them. Even for wallets that depend on a full-node for their information and network-access, there is the obvious dependence on a third-party, and the risk of identity-tracking - both of which are contrary to the ethos on which cryptocurrencies were created. This is the very reason that hardcore advocates of Bitcoin and Ethereum in particular, and cryptocurrencies in general, recommend running a full-node as your "wallet" - a node running the full reference client software, plus the complete blockchain validation, relaying, and optionally,

- **Custom/Proprietary Wallet Software in Exchanges:** As mentioned previously, a wallet in general refers to the entire mechanism employed to store and transact cryptocurrency and digital assets on blockchains like Bitcoin, Ethereum, Litecoin etc. As such, for centralized exchanges that hold and transact cryptocurrencies on behalf of their customers, customized implementations of wallets exist that are not transparent to the end users. This customized software, while following the fundamental guidelines outlined by the reference clients for particular blockchains, is subject to a lot of technical differences based on the unique design of every exchange. Thus, the mediation that such software undertakes between the central exchange and different public blockchains, is an area that requires careful cybersecurity consideration. An illuminating example in this regard, was the transaction malleability attack on Mt.Gox in 2014. The attack is a lesson in proper design of exchange-end software that interacts with the public blockchains. The attackers used transaction malleability to fool Mt.Gox's wallet software during withdrawals. It worked as follows:

    - Attackers would send a withdrawal request.
    - Mt.Gox would send a transaction to the Bitcoin network transferring the requested amount.
    - Attackers would change the TXID through transaction malleabilit.
    - The changed TXID would be approved (in many, if not most, cases) and the BTC would be transfered.
    - Mt.Gox would still see the original TXID that it was tracking (the flaw in the design), and decide that it was unconfirmed.
    - Attackers would request another withdrawal for the same BTC, and as per Mt.Gox's state, as the transaction has not gone through, it would send a new transaction - giving the attackers another transaction to tamper with.
    - The process was repeated until a significant number of BTC was stolen, before Mt.Gox realized there was a problem - that too owing to the numerous complaints of delays by Mt.Gox customers.

While this malleability weakness in the Bitcoin protocol has been addressed, the entire fiasco serves to shed light on a more general issue: the code and security design of software that interacts with public blockchains is still in its early days when compared to established domains, and thus requires a substantial focus on a holistic appreciation of cybersecurity in this new age of blockchain technology

While not an exhaustive list, these three fundamental parts of the exchange ecosystem provide the greatest insights into the current and future challenges that pertain to secure operation of blockchain-enabled products. It is very much possible to build blockchain-enabled solutions with as low a risk of cyber-attacks as any established Internet company, especially if the inherent security of blockchain is leveraged along with professional, specialized auditing and product design consultation.

Final Thoughts

The primary objective of focusing on cryptocurrency exchanges in this document, has been to analyze the most widely understood and used product built on and around blockchain technology. Most, if not all, of the lessons learnt by the blockchain community through experience with the functioning (and more importantly, dysfunctioning) of exchanges, are transferable to other blockchain ventures and products. This leads us to acknowledge the vital importance of specialized security audits of code, employment of best practices, and crucial product design decisions when it comes to successfully building and operating solutions powered by blockchain technology.

GENERAL INQUIRIES
contact@stratuscyber.com 443-292-2966

TECH SUPPORT
support@stratuscyber.com 602-900-9321