CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	A secure, distributed and scalable infrastructure for remote generation and use of cryptographic keys
Authors	Bareato, Claudio;Palmieri, Paolo;Regazzoni, Francesco;Venier, Oreste
Publication date	2020-09
Original Citation	Bareato, C., Palmieri, P., Regazzoni, F. and Venier, O. (2020) 'A secure, distributed and scalable infrastructure for remote generation and use of cryptographic keys', 2020 2nd Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS), Paris, France, 28-30 Sept. pp. 102-106. doi: 10.1109/BRAINS49436.2020.9223285
Type of publication	Conference item
Link to publisher's version	https://ieeexplore.ieee.org/document/9223285 - 10.1109/BRAINS49436.2020.9223285
Rights	© 2020 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works
Download date	2024-04-26 18:32:25
Item downloaded from	https://hdl.handle.net/10468/10673

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

A secure, distributed and scalable infrastructure for remote generation and use of cryptographic keys

Claudio Bareato
Independent Researcher
Lugano, 6900, Switzerland
claudio.bareato@bareato.ch

Paolo Palmieri
School of Computer Science
University College Cork
Cork, T12 YN60, Ireland
p.palmieri@cs.ucc.ie

Francesco Regazzoni
ALaRI, University of Lugano
Lugano, 6900, Switzerland
regazzoni@alari.ch

Oreste Venier
Independent Researcher
Lugano, 6900, Switzerland
oreste@drovenier.it

***Abstract*—This paper outlines an ad-hoc architectural design and a practical implementation of a secure, distributed, fault tolerant and scalable infrastructure comprised of a distributed network of electronics hardware systems that remotely generate cryptographic keys, store them, digitally sign cryptographic transactions based on such keys, and record the transactions on a blockchain. The proposed solution is suitable to service both crypto-finance and non-finance applications, and the physical infrastructure is designed to be implemented using off-the-shelf industrial electronics, which further sustains the already scalable-by-design infrastructure architecture.**

***Index Terms*—Blockchain, Crypto Assets, Physical Security**

I. Introduction

Since the introduction of Bitcoin in 2007 [1], the number of applications based on blockchain and crypto-assets paradigms has increased exponentially. A common requirement for these applications is the use of cryptographic assets, including private keys and derived concepts such as crypto-currency wallets. These digital assets are extremely vulnerable to theft, as proved by the number of cases where bitcoins for millions of dollars are stolen [2]. It is therefore crucial to secure these assets, while maintaining the ability to perform operations such as signing or currency transfers. A number of solutions have been proposed for this purpose. Mann and Loebenberger propose a two-factor authentication system for the Bitcoin protocol [3]. Wu et al. introduced a secure joint Bitcoin trading mechanism relying on partially blind fuzzy signatures [4]. Gentilal et al. focus instead on a system protection layer, and utilise the recent ARM extension of processors architectures, designated TrustZone, which allows for the separation of trusted and non-trusted environments [5].

In this paper, we propose the design of a novel distributed infrastructure, targeted at securely performing a full life-cycle management of the cryptographic assets that are instrumental to blockchain applications [6]. At present, such applications include tokenization of assets, notarization, smart contracts management, and all the operations involving crypto-currencies including the transfer of funds. We propose a complete hardware solution to remotely generate cryptographic keys, store them, digitally sign cryptographic transactions based on such keys, and record the transactions on a blockchain. Such infrastructure was developed to provide a concrete solution to a currently unmet practical need that is being exhibited by a fast-growing population of users who own cryptographic assets (such as secret keys and crypto-currency wallets), i.e. the need to simultaneously: 1. retain full control, throughout the whole lifecycle of the keys (generation, storage and usage), over the cryptographic private keys that provide unlimited access to the users' own cryptographic assets; 2. remotely perform cryptographic operations while at the same time preserving the security of the underlying keys.

The proposed solution includes the system architecture, the communication protocol and the keys management scheme. A dedicated approach for the hardware digital circuits have been developed to ensure that each system's core cryptographic engine, which stores the keys within a controlled physical perimeter, never gets connected to external networks, not even while the system is in operation. The cryptographic circuit is designed to be insulated from external networks at all times, no matter what the system state becomes. This constitutes a significant advantage compared to the other few existing automated solutions [7], [8], [9], [10], as the latter limit their operations to activating a cryptographic circuit and connecting it to the network(s) when needed to.

II. Related Works

At present, the only sector where the needs identified in the previous section have been fulfilled or at least addressed to some extent, is crypto-finance, in particular, the crypto-token and crypto-currency payment and investment applications. These applications deal with crypto-currencies or crypto-tokens intended as a subclass or specific type of the more general category termed crypto assets [11]. Systems designed for crypto-finance normally manage only a specific subset of cryptographic operations (typically related to crypto-currency wallets [12]). Solutions in this area are already being commercialised. For

978-1-7281-7091-6/20/$31.00 ©2020 IEEE

consumer-oriented applications, hardware wallets [7], [10] allow end users to perform financial transactions using crypto-currencies or crypto-tokens and to track currencies or tokens in their portfolios. They mostly operate in a stand-alone mode. The typical use case for these solutions is an individual end user who employs the hardware wallet connected to a dedicated software application. The functionalities include storing seed and cryptographic keys on the physical dongle, while the management of the wallet, i.e. the management of the various financial positions associated to the crypto currencies, is carried out by a software application running on a smartphone or a PC. On the business to business side, there are few implementations of hardware security modules (HSM) [13] dedicated to crypto assets, such as the one produced by Ledger. However, from the analysis of the publicly-available documentation such modules do not entrust end-users with a direct and full control over their cryptographic private keys [14].

There exist also a number of wallet that haven't yet achieved implementation. Recently, Chen et al. proposed a blockchain-based payment collection supervision system using pervasive Bitcoin digital wallet [15]. An older wallet implementation is BlueWallet by Bamert et al. [16], a proof-of-concept Bitcoin hardware token. The BlueWallet device communicates using Bluetooth Low Energy and securely sign Bitcoin transactions. The device can also be used as an electronic wallet in combination with a point of sale and serves as an alternative to cash and credit cards.

However, none of the presented solution proposes distributed architectures for the storage and management of cryptographic assets. In particular, a distributed privacy-preserving solution in this context, similar in nature to those proposed for other distributed systems [17], is missing. In this paper, we address this gap by designing a distributed architecture. Our approach enables remote management and storage of crypto-currency wallets and private keys while ensuring a high level of availability. To increase the security of the communication and to provide an additional privacy layer, a mix-style network with layered encryption [18], [19] is used.

III. System Design

We propose an architecture that achieves physical insulation of core cryptographic components and information (e.g. keys), targeted at providing security against adversaries with remote access to the device. The proposed architecture has been implemented in a prototype, described in Section IV.

The infrastructure we propose combines layered encryption and hardware security modules and it is composed by three classes of systems: *user devices*, *platform nodes* (also called relays), and *remote nodes*. The capabilities of each class of nodes, as well as the information they possess is detailed in Table I.

User devices U_i The user devices are designed to be small portable devices, able to perform simple operations when connected to a host computer. Their role is to authenticate the user, and enable communication with the platform nodes. Each user has a single device. The device itself stores on protected memory an elliptic curve public/private key pair $(\mathsf{pk}_{U_i}, \mathsf{sk}_{U_i})$ based on Curve 25519 [20]. Through a software to be installed on the host computer, the user device is able to establish an encrypted connection using ECC and the key pair, as well as symmetric cipher Salsa20 [21]. We note here that attacks on the host computer are outside the scope of this paper, and are not considered in the analysis.

Platform nodes P_i Platform nodes act as relays interposing the communication between the user devices and the remote nodes. Their role is to separate remote nodes from the Internet, and provide access control and – in commercial implementations – user accounting and billing. A number of relays are present, depending on the size of the overall infrastructure. The nodes are also protected from DDoS attacks through commercially available cloud solutions. Each node has an elliptic curve public/private key pair, and is capable of establishing an encrypted connection using ECDH and a symmetric cipher Salsa20, similarly to the user devices. Platform nodes also store a number of time-stamped nonces that have been created by each remote node for later use in the establishment of an Elliptic-Curve Diffie-Hellman (ECDH) key agreement. The nonces are signed by the generating node with its private key.

Remote nodes D_i Several remote nodes collectively store the cryptographic assets (such as private keys) that are guarded by the infrastructure. In order to provide a level of isolation, the assets are stored on a dedicated device that is distinct from the main remote node system, connected to the Internet – although protected by a firewall which allows incoming connection from platform nodes only. The dedicated electronic device has been conceived to remain offline most of the time, and is activated only when it is necessary to perform relevant cryptographic operations as instructed by the user devices. This separate cryptographic device is designed to generate, store and use cryptographic keys. As all the relevant cryptographic operations can be performed internally to the node, the keys are never exposed to the main node system, nor to the outside world. In order to generate strong keys, the device embeds a true random numbers generator subsystem (TRNG).

Communication between the parties is described in Figure 1. The first step is taken by the user U_i, who initiates the communication by performing an Elliptic-Curve Diffie-Hellman (ECDH) key agreement with a selected platform P_i, which the user can choose freely among the available nodes. The agreed key is then used for encrypting any further communication between the two parties using the Salsa20. Once an encrypted link between U_i and P_i is established, the user selects a D_i he wants to use to

Devices	Keys and Information	Main Functions	HW Details		Web Interfaces

U_i	sk_{U_i}
	pk_{U_i}
	$\mathsf{pk}_{P_{1..n}}$
	$\mathsf{pk}_{D_{1..n}}$
		`Curve 25519`
		Signatures
		Verification
		`Salsa20`
			microcontroller
			display
			single button
			self-installing SW for interfacing with PC		

P_i	sk_{P_i}
	pk_{P_i}
	$\mathsf{pk}_{U_{1..n}}$
	$\mathsf{pk}_{D_{1..n}}$
	Ordered cache of $D_{1..n}$ nonces with expiration time
		`Curve 25519`
		Signatures
		Verification
		`Salsa20`
		`Lamport Signatures`
			HW module connected to server via USB		`ssh`

D_i	sk_{D_i}
	pk_{D_i}
	$\mathsf{pk}_{U_{1..n}}$
	$\mathsf{pk}_{P_{1..n}}$
	Asset to be protected
		`Curve 25519`
		Signatures
		Verification
		Key generation
		ECDSA
		`Salsa20`
		`Lamport Signatures`
			TRNG
			HW cryptography
			Shared memory
			Connected Part
			`Curve 25519`
			Verification
			`Salsa20`
				Not Connected Part
				`Curve 25519`
				Verification
				`Salsa20`
				TRNG
				ECDSA
					`ssh`

TABLE I: Functions of each component of the systems.

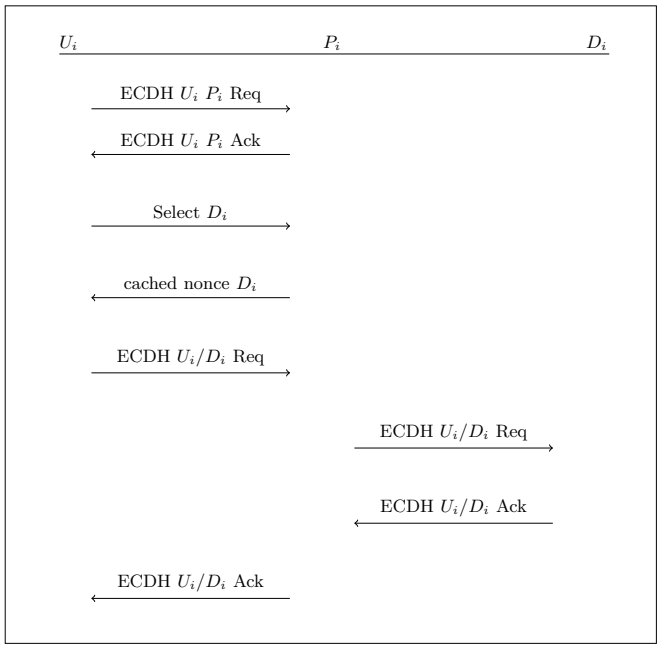


Fig. 1: The layered encryption protocol used to establish a secure tunnel between the user devices U and the remote nodes D, using platform nodes P as transparent relays.

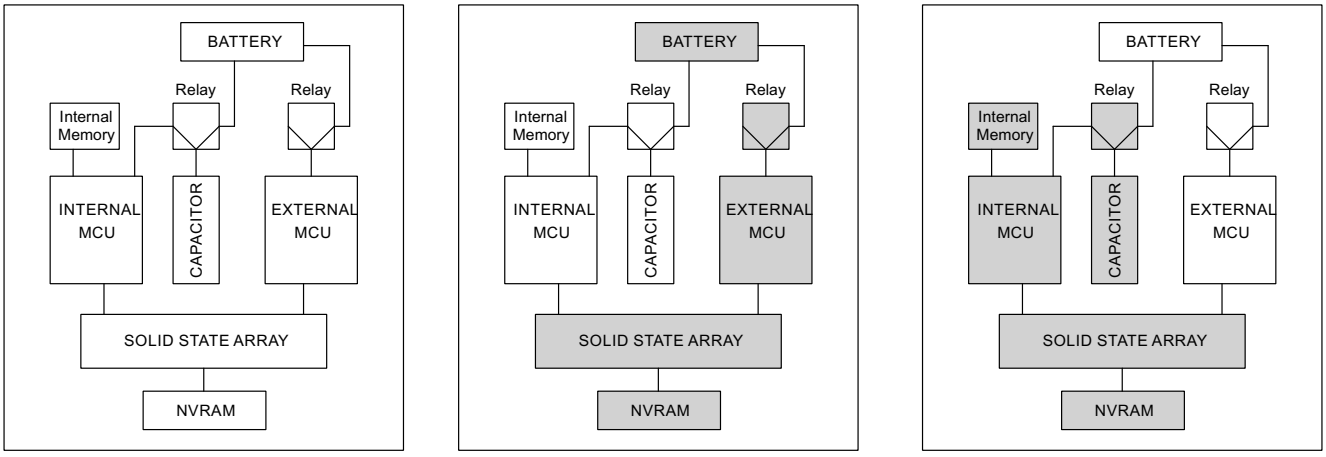
perform the required operations, and communicates his choice to the platform node. The platform node P_i selects a nonce precomputed by D_i among those that it stores cached, and sends it over to U_i. The user is able to verify the authenticity and integrity of the random nonce by checking D_i's signature against its public key. The nonce will allow U_i and D_i to perform an asynchronous Elliptic-Curve Diffie-Hellman key agreement using P_i as relay. Once a key has been agreed by U_i and D_i, an encrypted tunnel between them using the Salsa20 is established. The communication will however still be relayed by P_i, as U_i cannot communicate directly with D_i. Packets originating from U_i are therefore encrypted twice, using layered encryption in a way that is similar to that of onion routing [22]. The channel between U_i and D_i is used to transmit the cryptographic instructions to be performed using the crypto-assets stored on the remote node, and to communicate back to the user the results if necessary. We note here that P_i and D_i are also connected by an encrypted tunnel established using the same mechanism as in the case of U_i and P_i: this part of the protocol has been omitted from Figure 1 for simplicity.

IV. System Implementation

Figure 2a depicts the simplified version of the circuit implementing the nodes D_i. The system is composed of two microcontroller units (MCU), one dedicated to the communication (External), and one dedicated to the cryptographic operations. The architecture is conceived to insulate the part handling the communication from the part executing cryptographic operations using the protected asset.

Figure 2b and Figure 2c show this insulation. On the left, in Figure 2b, the box indicated the modules which are active during the communication with the external world. During regular operation, this part is active and the second part, the one devoted to computation of cryptographic operations, is switched off. Relays are used to enforce this insulation. When the cryptographic module is operating, as shown in Figure 2c, the part which communicate with the rest of the world is powered off. The cryptographically secure module is operating on an internal memory not directly accessible from other routine.

The two modules are insulated by means of a relay. Initially, the two relays are in a default condition. The microcontroller of the module communicating with the external world is powered by battery, while the microcontroller dedicated to cryptographic operation is completely



(a) General System (b) Active during communication (c) Active during cryptographic operations

Fig. 2: Block diagram of D_i components highlighting the part active during communication and during cryptographic operations.

switched off. When data are ready in the non volatile RAM (NVRAM), that in our case is implemented using a Ferromagnetic RAM (FRAM), and there is the need to carry out cryptographic operations securely, the internal module is switched on, and the external one is switched off. The secure module takes the power from a capacitor, to insulate it from the general power supply. The only common parts is the NVRAM, which is used for exchanging data to the cryptographic engine. However, the power supply of the NVRAM is connected to one source when it operates in one case, and to another source when it operates in another case. The NVRAM has a simplified serial interface which is designed to ensure insulation between the external world and the cryptographic module.

Below, we list the main components of the solution. A battery is the primary source of power. A microcontroller (called external MCU) holds the data to be processed (e.g. the cryptographic operations to be calculated) is powered through a relay. Then there is a supercapacitor (of about some dozen Farads) which is connected to another relay. Another microcontroller (called internal MCU) will execute the cryptographic operations using an internal memory which holds the user's key material. This memory use the same power line (not showed here) as the internal microcontroller. There is a high endurance ferromagnetic memory, NVRAM, and a solid state relay array (SSRA). A Complex Programmable Logic Device (CPLD) switch (which is purely passive) takes care of switching all the relays to one side or to the opposite, as further specified, through the control signals. NVRAM is connected through a Serial Peripheral Interface (SPI) both to the internal MCU and to the external MCU through SSRA (to one or to the other alternately). CPLD switch is connected through a serial Universal Asynchronous Receiver Transmitter (UART) control channel both to the internal MCU and to external MCU (also switched by SSRA).

Initially the relays are in default condition: external MCU is powered by the battery; the supercapacitor is charged and backed-up by the battery; the NVRAM and CPLD are powered through the battery; the NVRAM is connected to the external MCU through SSRA; external MCU can communicate commands to CPLD switch. In this phase the internal MCU is completely shut down, because its power supply is unconnected thanks to the first relay, and its buses are unconnected because of SSRA. Still in this phase, external MCU loads into the NVRAM the operations to be performed. When the process finishes, the CPLD switch is notified to commute both relays and SSRA. At this point, external MCU is shut down using the second relay, whilst the supercapapacitor, that is charged, powers up: internal MCU via the first relay, and CPLD and the switch and the MVRAM (without discontinuance) through direct supercap connection. In this second phase, internal MCU loads from NVRAM the operations to be performed, it executes them and it updates the NVRAM. When it finishes, it informs the CPLD switch to commute both relays, then the SSRA and therefore the internal MCU is again shut down and isolated, while the external MCU becomes active. A functioning prototype is visible in Figure 3.

V. Conclusions and Discussion

This paper outlines a novel distributed architecture and communication protocol for the management and storage of cryptographic assets, including secret keys and cryptocurrency wallets. The proposed infrastructure is a distributed system using a layered encryption communication protocol. The cryptographic private keys that control the associated crypto-assets are split among a number of different nodes, physically located far apart from each other, providing the intrinsic redundancy of distributed systems.

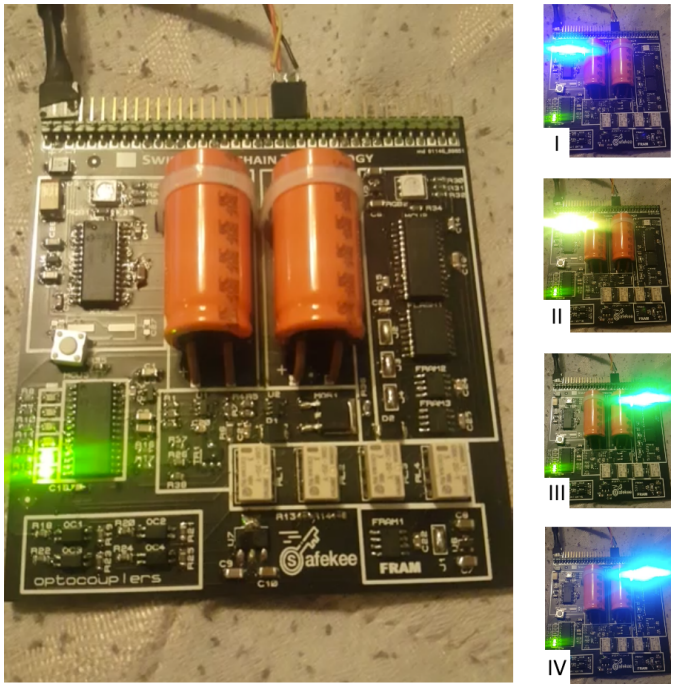


Fig. 3: The functioning prototype. On the right, the different phases can be seen via LED indicators: I) the external communication side is active, and the transaction is received; II) the transaction is stored on NVRAM; III) The communication side is inactive, and the cryptographic side is activated, transactions are loaded from NVRAM; IV) after signing, the transaction is stored on NVRAM.

Further protection of the assets is provided by an isolated electronic architecture that allows to securely perform a variety of cryptographic operations remotely. A dedicated mechanism has been conceived to insulate the hardware cryptographic unit in remote nodes, and to make it active only when needed. The conceived architecture constitutes an extension of the "cold storage" concept. The remote node devices not only store cryptographic material, but also generate the private keys internally, therefore never exposing them to the outside world.

This system is oriented towards physical security rather than processing speed. Normally, a digital signature transaction system is able to develop a very high number of signatures per second because it communicates directly with the outside world. The system described here requires instead a series of additional steps in order to enable isolation of the cryptographic part, such as the loading of the supercapacitor; and the presence of switched ferromagnetic memories, which need to be read and written by both the external and internal processor. For this reason, in a typical usage session that includes charging the supercapacitor, managing previously signed transactions and receiving, depositing, signing and subsequent writing of new transactions, it is possible to sign a few hundred transactions per minute.

The functional characteristics of the proposed solution address several limitations of current technologies, and in particular allow end users to retain direct operational control over their private keys. Since the control protocol used to instruct operations is abstracted from any specific type of message, the infrastructure can also be extended to support future applications that require standard cryptographic functions, including solutions that go beyond crypto-currencies. We believe that the proposed approach is cost-effective and practical, and represents a fundamental step towards a new generation of secure devices enabling the remote storage and management of cryptographic assets.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[2] S. T. Ali, D. Clarke, and P. McCorry, "Bitcoin: Perils of an unregulated global p2p currency," in *Cambridge International Workshop on Security Protocols*. Springer, 2015, pp. 283–293.

[3] C. Mann and D. Loebenberger, "Two-factor authentication for the bitcoin protocol," *Int. J. Inf. Sec.*, vol. 16, no. 2, pp. 213–226, 2017.

[4] Q. Wu, X. Zhou, B. Qin, J. Hu, J. Liu, and Y. Ding, "Secure joint bitcoin trading with partially blind fuzzy signatures," *Soft Comput.*, vol. 21, no. 11, pp. 3123–3134, 2017.

[5] M. Gentilal, P. Martins, and L. Sousa, "Trustzone-backed bitcoin wallet," in *Proceedings of CS2@HiPEAC 2017*, 2017, pp. 25–28.

[6] M. Crosby, P. Pattanayak, S. Verma, and V. Kalyanaraman, "Blockchain technology: Beyond bitcoin," *Applied Innovation*, vol. 2, pp. 6–10, 2016.

[7] "Ledger," https://www.ledgerwallet.com/, accessed: 2018-03-1.

[8] "Casa," https://keys.casa, accessed: 2018-03-1.

[9] "Bitgo," https://www.bitgo.com/, accessed: 2018-03-1.

[10] "Trezor," https://trezor.io/, accessed: 2018-03-1.

[11] J. P. Conley *et al.*, "Blockchain and the economics of crypto-tokens and initial coin offerings," Vanderbilt University Department of Economics, Tech. Rep., 2017.

[12] R. Gennaro, S. Goldfeder, and A. Narayanan, "Threshold-optimal DSA/ECDSA signatures and an application to bitcoin wallet security," in *14th ACNS 2016.*, 2016, pp. 156–174.

[13] S. Mavrovouniotis and M. Ganley, "Hardware security modules," in *Secure Smart Embedded Devices, Platforms and Applications*, 2014, pp. 383–405.

[14] E. Hofmann, U. M. Strewe, and N. Bosia, *Concept—Where Are the Opportunities of Blockchain-Driven Supply Chain Finance?* Cham: Springer International Publishing, 2018, pp. 51–75.

[15] P. Chen, B. Jiang, and C. Wang, "Blockchain-based payment collection supervision system using pervasive bitcoin digital wallet," in *WiMob 2017*, 2017, pp. 139–146.

[16] T. Bamert, C. Decker, R. Wattenhofer, and S. Welten, "Bluewallet: The secure bitcoin wallet," in *STM 2014.*, 2014, pp. 65–80.

[17] N. Zeilemaker, Z. Erkin, P. Palmieri, and J. A. Pouwelse, "Building a privacy-preserving semantic overlay for peer-to-peer networks," in *WIFS 2013*. IEEE, 2013, pp. 79–84.

[18] P. Palmieri and J. A. Pouwelse, "Key management for onion routing in a true peer to peer setting," in *IWSEC 2014*. LNCS, vol. 8639, Springer, 2014, pp. 62–71.

[19] T. Güneysu, F. Regazzoni, P. Sasdrich, and M. Wójcik, "THOR - the hardware onion router," in *FPL 2014*, 2014, pp. 1–4.

[20] D. J. Bernstein, "Curve25519: New diffie-hellman speed records," in *PKC 2006*, 2006, pp. 207–228.

[21] ——, "The salsa20 family of stream ciphers," in *New Stream Cipher Designs - The eSTREAM Finalists*, 2008, pp. 84–97.

[22] P. F. Syverson, D. M. Goldschlag, and M. G. Reed, "Anonymous connections and onion routing," in *S&P 199*, 1997, pp. 44–54.