Improving WebRTC Security via Blockchain Based Smart Contracts

Berat Yilmaz
Defence Systems Technologies
Aselsan Inc.
Ankara, Turkey
beyilmaz@aselsan.com.tr

Ertugrul Barak
Defence Systems Technologies
Aselsan Inc.
Ankara, Turkey
EBarak@aselsan.com.tr

Suat Ozdemir
Computer Engineering Department
Gazi University
Ankara, Turkey
suatozdemir@gazi.edu.tr

***Abstract*—WebRTC (Web Real-Time Communication) is one of the most popular technologies that provide the infrastructure for real-time data communications. Many applications used in military and civilian life depend on WebRTC technology directly or indirectly in the background. Due to its wide area of applications, the security of WebRTC communications is of great importance. In this paper, we improve the existing security infrastructure of WebRTC. The proposed solution provides a real-time communication infrastructure between peers on the network by making use of quite new and popular blockchain and smart contract technologies. Within the scope of the paper, a web-based application with video calling feature demonstrating blockchain interactions and suggested security enhancements have been developed. In the application, operations such as adding, updating, and authorizing users can be performed on the smart contract deployed on the live Ethereum network. Based on the principle of immutability of blockchains, users on the smart contract are mapped to the peers (nodes) in the WebRTC communication to be established and used in the identity verification process. Consequently, this paper proposes a secure identity verification method that could be used by real-time WebRTC communication peers to verify each other on the blockchain. The proposed solution is shown to resilient against man-in-the-middle attacks.**

Keywords—WebRTC, Blockchain, Ethereum, Smart Contract, Real-Time Communication Security

I. INTRODUCTION

In real-time communication, the content of the transferred data can vary from casual communication to military or state secret information. When it comes to standard usage scenarios, data security remains a feature that can be ignored, except for specific topics like banking activities and medical records. However, when it comes to transferring confidential or critical data between institutions, organizations, or critical persons, ensuring the security of the data is critical. Most advanced authentication mechanisms can provide security in all usage scenarios; however, it can be taxing on the system both in terms of cost and performance, and hence it may not be feasible or practical. In this study, the main emphasis is put on the security aspect of the data transmitted in real-time rather than other issues. Besides, it was taken into consideration who transmitted these data. As the real-time communication library, the proposed method is built on WebRTC application technology, which is open-source and does not need the installation [1].

In this study, while offering a solution to improve the security of WebRTC-based real-time communication, today's quite popular technology blockchain was used. The re-popularization of blockchain is directly related to the Bitcoin cryptocurrency, which also severely affected the global economy. The increase in the number of academic studies with blockchain content in recent years is directly related to the article titled "Bitcoin: A Peer-to-Peer Electronic Cash System" [2] published in 2008. This popularization has led the researchers to think in detail about blockchain technologies and, as a result, has revealed many new research and application areas.

Smart contracts are one of the most critical application areas that arise with the popularization of blockchain. While Bitcoin provides a limited level of smart contract support with its scripting language, Ethereum emerged in 2014, and the smart contract technology was brought to a new level [3]. Without a central authority, features such as the identification of digital contracts between different stakeholders and the automatic implementation of targeted actions according to the outcome of the contract have made smart contracts among the most promising application areas of blockchain technology [4].

In this study, the aim is to improve the level of security provided for a defined communication group in projects using WebRTC based communication infrastructure. For this purpose, a group is created by bringing together a certain number of predefined users, who are authenticated via the proposed blockchain-based authentication method. A solution is proposed such that only previously authorized users can communicate with each other on the WebRTC infrastructure.

The rest of the paper is organized as follows. Section II provides an overview of the related work, while Section III presents the problem statement. Section IV discusses WebRTC Security, and Section V mentions the basics of the blockchain technology. Section VI explains the proposed solution step by step in detail, and Section VII discusses the performance evaluation of the proposed protocol. Finally, Section VIII evaluates the obtained results and interprets the outcomes of the study.

II. RELATED WORK

Some important studies in the literature where WebRTC application technology and blockchain are used to increase security are summarized in this section.

In the study titled "WebRTC Based Augmented Secure Communication" [5], a method for providing end-to-end WebRTC-based secure communication is proposed. In the proposed method, the parameters for key exchange in standard WebRTC signaling are securely transferred by public key-based cryptography methods that work on smartcards. In the study, it is expressed that the method is applied in a real environment and shown that it is a feasible solution.

In the study titled "Security of WebRTC" [6], current WebRTC structure and security; is reviewed in the contexts of communication disruption, modification, and eavesdropping.

978-1-7281-5628-6/20/$31.00 ©2020 IEEE

Also, this study examines WebRTC security by simulating real WebRTC environments and attacks over several representative scenarios.

In the study titled "Security Mechanisms for Signaling in WebRTC-Based Peer-to-Peer Networks" [7], there is a mechanism called Chord used to establish a peer-to-peer network on WebRTC. "Chord" is an efficient and well-known way to create an overlay for a structured peer-to-peer network. However, "Chord" does not have secure authentication and confidentiality mechanisms. For this reason, the man-in-the-middle attack can occur while the peers agree on the WebRTC parameters for direct connection. This vulnerability is solved by a proposed hybrid encryption method: In this method, each host generates a unique long-term asynchronous key pair for authentication and short-term asynchronous key pairs to establish synchronous secret keys. Using these key pairs, peers can exchange WebRTC connection parameters via end-to-end authenticated and encrypted messages.

In the study titled "Usable Authentication Systems for Real-Time Web-Based Audio/Video Communications" [8], it is expressed that WebRTC is a new multimedia library, which provides real-time multimedia communication via browsers, but it lacks an authentication system. Solutions that use third-party services from Google, Facebook, etc. as identity providers are said to be limiting. In this study, a flexible authentication system solution was developed for a web-based multimedia communication system using the WebRTC library and the Google channel API. This solution has a user management software framework that includes various authentication scenarios, including local and cloud-based authentication (e.g., Google, Facebook login). In the study, after presenting different authentication scenarios, the degree of usability of the solutions is investigated, and different authentication solutions are proposed considering the security requirements in terms of security level or user convenience and preferences.

In the study titled "WebRTC: Your Privacy is at Risk" [9], WebRTC, which enables direct communication between browser-to-browser or device-to-device, is described as the next step towards the elimination of current browser limitations. The study focuses on security and privacy, which affects the implication imposed by this emerging technology. Besides, various attacks have been developed in which the security and privacy of the actual user are questioned, as well as the privacy of other devices on the same network. As a result of the evaluations, even though WebRTC is based on a robust security foundation, privacy of user and communication security can be compromised due to several design decisions.

In the thesis titled "The Use of Blockchain in Digital Identity Management Systems for KYC (know your customer) Processes in Banks and Its Evaluation from Cybersecurity Perspective" [10], blockchain is used to verify digital identities. It has been mentioned that digital identities can be used to solve problems with physical identities and inefficiencies in know your customers. In this thesis, one of the usage models of blockchain for use in KYC processes is discussed in detail, and cybersecurity risks are defined, and suggestions are proposed.

In the study titled "Using Blockchain for IOT Access Control and Authentication Management," [11] a blockchain-based solution that allows for authentication and secure communication to IoT devices is proposed. The proposed solution benefits significantly from the intrinsic features of blockchain and also builds on existing authentication schemes. Also, this blockchain-based solution, architecture, and design allow for accountability, integrity, and traceability with tamper-proof logs.

In the thesis titled "Using Blockchain Technology and Smart Contracts for Access Management in IoT Devices" [12], the blockchain technology is examined and also studies access control potential of IoT devices. In this thesis, a proof of concept system has been designed and implemented that demonstrates a simplified access management system using the Ethereum Blockchain.

The literature survey shows that there are many active research topics related to WebRTC security and blockchain. Many active studies appear to focus on the lack of a standard authentication system in WebRTC. Examination of studies to improve the security of a system using blockchain-based smart contracts indicates most of the limited numbers of studies found are aimed at solving security problems in the IoT area or improving existing security solutions.

III. Problem

When using the DTLS-SRTP security method recommended by WebRTC, a situation occurs where users are unable to authenticate their remote peers. This circumstance causes many security problems, especially the man-in-the-middle (MiTM) attacks.

IV. WebRTC Security

In a WebRTC-based real-time communication, whether the communication is secure or not is a fundamental issue to deal with. The criticality of the issue depends on the application area, and it is critical, while, especially considering the areas where the security of the transmitted data is extremely high, such as military communication.

When security is examined on the peers that will communicate with WebRTC, one of the requirements for security is that microphone and camera accesses are not performed without the permission of the user. This access takes place only with the permission of the related WebRTC peers. Also, even in cases where this approval is provided, the activity status of the microphone or camera must be informed to the peer simultaneously [13].

All steps to establish a peer-to-peer connection in WebRTC are performed on one or more specially selected servers. While these servers are called signaling servers, the created mutual communication line is called as signaling channel. Choosing the signaling server and utilization of the signaling channel are entirely up to the choice of application developers.

Peers are expected to transmit the packages, called Session Description Protocol (SDP) [14], which contain the parameters of the real-time communication to be established, to each other over the signaling channel. Also, choosing protocols (SIP, REST, WebSocket, XMPP) used while transmitting over the signaling server is entirely up to the choice of application developers.

In the secure media channel to be allocated, there are two standard methods that application developers can determine

how the symmetric key is transmitted between peers, and these are namely SDES [15] and DTLS-SRTP [16].

In the SDES approach, the symmetric key to be used in the SDP packages is transported openly (cleartext), so it is not considered as reliable today. For this reason, it is not supported by any browser other than Google Chrome, which continues its support due to its backward compatibility.

In the DTLS-SRTP approach, certificates are exchanged between the peers, as in the standard TLS [17]. These self-signed certificates are transferred between peers on the media channel, without being signed by any certification authority (CA). The fingerprints of the certificates are sent mutually between peers using SDP packets via the signaling channel. Thus, the peers who will communicate mutually in real-time have both certificates and fingerprints of each other. Now, it can be confirmed that the peer who created and sent the certificate is the same [18, 19]. Although the method eliminates the trust in the signaling server, it cannot provide a solution against the man in the middle (MiTM) attacks, since it cannot provide authentication. As shown in Figure 1, there is a Z(hacker) user who sniffs the network traffic in an example, and X and Y users (peers) are trying to communicate with each other. Hacker Z treats X as Y, Y as X. So, Z can sniff all communication data.

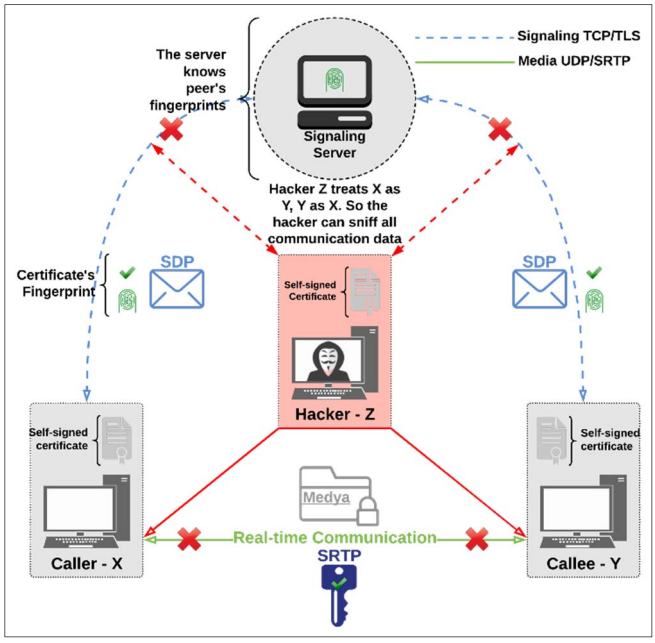

Fig. 1. MiTM attack in the DTLS-SRTP approach.

V. BLOCKCHAIN

Blockchain is an encrypted, distributed, interconnected, and continuously growing database used to store transaction records that allow the reliable transfer of Bitcoin and other cryptocurrencies. This structure can be seen in Figure 2, and in this Figure, the blocks are connected like chainrings to another block. The decentralized (distributed) structure of this database prevents the database from manipulation of a centralized particular group or community. In the blockchain, devices are directly connected like the P2P network; thus they are kept up to date and maintain their synchronization. In the blockchain, the data added to blocks cannot be changed and deleted (immutability feature). The blockchain continues to grow and strengthen with each block added to it [20].

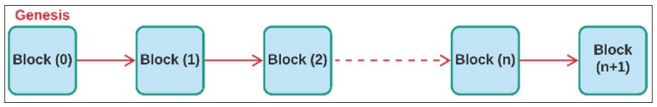

Fig. 2. Blockchain block sequence.

A. Ethereum

Ethereum is a programmable blockchain platform designed in 2015 by a young genius named Vitalik Buterin, at the age of 19, imagining much more than a cryptocurrency. Although its status is generally seen as a sub-cryptocurrency, Ethereum is essentially an innovative system aimed to develop blockchain technology more and to use of it in more areas. In the Ethereum network, each system runs a virtual machine called Ethereum Virtual Machine (EVM). These virtual machines allow any application written in programming languages provided by Ethereum (Solidity, Viper, Serpent, etc.) to be run. EVM makes the process of creating blockchain applications easy and efficient. Ethereum enables thousands of different applications to be developed on a single platform instead of building a completely original blockchain for each new application. Owing to the language structures offered by EVM, all cases that are observed theoretically can be prepared as a program in Ethereum [22-25].

B. Smart Contracts

The smart contract can be defined as the transactions that their rules have already been determined, and if the parties fulfill these rules, these transactions will start to run automatically [21]. The concept of smart contract was first introduced in 1994 by Nick Szabo, a computer scientist, and mathematician, without using blockchain expression [26]. When Vitalik Buterin's work titled Ethereum was published in late 2013, [23], Buterin used the concept of "Smart Contract" when describing the applications in this work. Then this concept was frequently used in blockchain technology. Smart contracts can be defined as the following [27]:

- A block of computer code in which logical flows are pre-written,
- It can be stored and spread on a distributed, decentralized platform (blockchain networks),
- It can be executed and operated by a computer network (the computers that comprise the blockchain),
- Its reliability can be verified by a computer network (blockchain network),
- It can be defined as small programs that can lead to updates (crypto money payments/transfers, creating new smart contracts) on the structure or platform on where it is located.

The defined smart contracts above are shown in Figure 3.

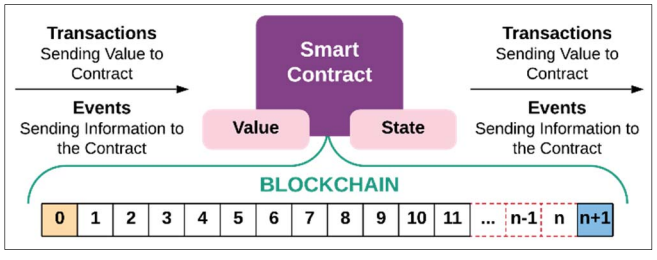

Fig. 3. Smart contract.

VI. Proposed Solution

WebRTC application technology is used extensively today. As a result of this intensive use, numerous researches are being conducted, especially on security issues. Although WebRTC is a security-oriented library, the results of these studies show that there are some weaknesses in security. Within the scope of the paper, it is focused on the most important of these weaknesses and suggested solutions. When implemented with the DTLS-SRTP security approach, WebRTC looks very secure in terms of established real-time communication. However, it comes to the process of initiating the communication (handshake steps); some security gaps attract attention. To explain these security issues in detail, communication between peers in WebRTC is safe, but it cannot be verified from the identities of the peers in communication with standard WebRTC library's methods. WebRTC offers different methods for solving this situation, but the decision whether to apply it or not is left to the application developers. Behind each proposal, there are central third-parties that are needed to be trusting, and trust in these third-parties also poses a separate security challenge.

Within the scope of this study, the solution proposed that enables mutual authentication of colleagues who will communicate with WebRTC. Thus, in real-time communication, peers can continue their communication by making sure of each other's identities. Different from WebRTC's current security approach, in the proposed solution, the peers who will communicate, do not need to trust solely one central mechanism or authority. Within the scope of this solution, a smart contract has been designed that includes user information and permissions and allows various operations on these. The designed smart contract is deployed to the distributed (decentralized) Ethereum Test Network, whose name is Rinkeby. This deployed smart contract is implemented for authentication in the WebRTC handshake steps.

A. User-Fingerprint Mapping

In WebRTC, the DTLS-SRTP security approach is the default and recommended security approach. In this work, a security development based on the DTLS-SRTP security approach is proposed. In this approach, for each peer to communicate, a self-signed certificate is generated. This certificate is utilized to set up a secure DTLS tunnel where the symmetric key of the real-time communication will be exchanged (See Figure 4).

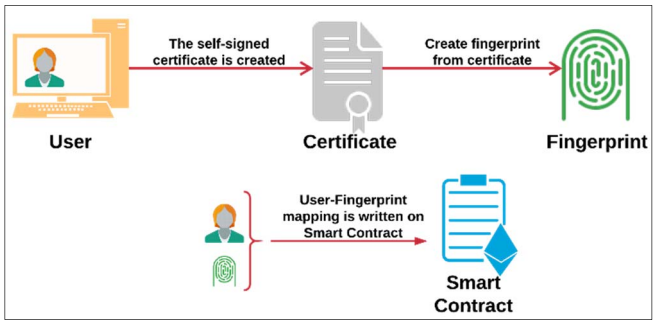

Fig. 4. User-fingerprint mapping steps.

B. Preparation of the Smart Contract

Inside of the smart contract which coded with the Solidity programming language, a struct definition is described. This struct has fields that are characterizing peers who will communicate. These fields are identity (id), name, surname, address that identifies the user in the Ethereum Word, and fingerprint information of the user's certificate, which is very critical for the solution.

This written smart contract includes methods such as adding users, updating the user's information, editing the communication authority, and questioning the communication authority of the user via fingerprint. Although these methods are defined on a smart contract on the blockchain network, only authorized peers can access the methods. In other words, not all methods can be operated by ever peer. Owing to smart contracts' authorization features, method calls can be restricted, thus only intended users can call specific methods (See Figure 5).

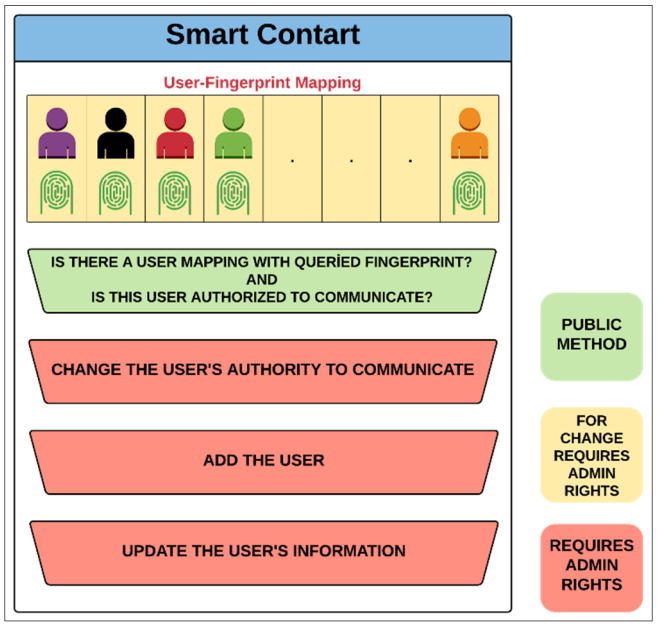

Fig. 5. Scheme of the smart contract.

C. Specifying the Communication Group

In the proposed approach, two levels of authorization are offered for the users. These levels are admin and standard, respectively. The authorization levels of users cannot be changed during the application life cycle. Only the user with admin authority can perform all functions requiring authorization. The users with standard authority are not authorized to perform these functions, which are requiring authorization. Additionally, the users with standard authority are not authorized to perform these functions, which are requiring admin authority. The standard authorized users correspond to peers who can communicate with each other on the proposed solution via WebRTC. The admin can be one of the parties to communicate with each other by acting as a standard user in the system. Within the scope of this paper, it is accepted that the person who deployed the contract to the Ethereum network and the admin is the same. The admin's main tasks are listed below:

- Deploying prepared, smart contract at the beginning of the process,
- Specifying the communication group,
- Being able to update the information of users in the communication group and add a new one at any time,

- Being able to change any standard user's authority to communicate at any time (such as "can communicate" or "cannot communicate")

D. Deploying Smart Contract

In order for the prepared contract to become available, it must be deployed into the relevant blockchain network ("Rinkeby" for this work). This deployment can only be applied by the user with admin authority. Before the contract is deployed to a live Ethereum network, all the features it offers should be checked with extensive tests, and if any bugs found in this process, they should be corrected. The smart contract, which passes all these controls and test processes, can be uploaded to the live Rinkeby test network [28].

E. Using Smart Contract as an Identity Provider

While using the DTLS-SRTP security approach, it is assumed that X (caller) and Y (callee) peers, who want to communicate with each other, have generated their self-signed certificates, before the SDP packets start to be exchanged. Likewise, it is assumed that fingerprints belonging to these certificates are written on the smart contract on the blockchain by making necessary matches by the admin. Finally, the authority to communicate of X and Y peers is changed to "can communicate" by the admin. Then the following steps are run, respectively:

1) X inserts its certificate's $fingerprint_x$, which is created previously and will be used to set up a safe DTLS tunnel, in the SDP_{offer} package.
2) The SDP_{offer} package is finalized by X.
3) The SDP_{offer} packet is sent to the signaling server over the signaling channel by X.
4) The SDP_{offer} packet is sent to Y by the signaling server.
5) Y gets the $fingerprint_x$ found in the SDP_{offer} from X. Y queries whether there is a peer match with the queried $fingerprint_x$ in the Ethereum blockchain or not. If exists, Y queries whether this peer (X) is authorized to communicate or not.
6) The previous step is considered unsuccessful if either the peer (Y) is not defined on the smart contract, or the peer (Y) exists but it is not authorized to communicate. In both cases, the authentication is considered unsuccessful, and session initiation (handshake steps) is interrupted. If the previous step completes successfully, X is authenticated, and the process continues with the next step.
7) Y inserts its certificate's $fingerprint_y$, which is created previously and will be used to set up a safe DTLS tunnel, in the SDP_{answer} package.
8) The SDP_{answer} package is finalized by Y.
9) The SDP_{answer} packet is sent to the signaling server over the signaling channel by Y.
10) The SDP_{answer} packet is sent to X by the signaling server.
11) X gets the $fingerprint_y$ found in the SDP_{answer} from Y. X queries whether there is a peer match with the queried $fingerprint_y$ from the Ethereum network or not. If exists, X queries whether this peer (Y) is authorized to communicate or not.
12) The previous step is considered unsuccessful if either the peer (X) is not defined on the smart contract, or the peer (X) exists, but it is not authorized to communicate. In both cases, the authentication is considered unsuccessful, and session initiation (handshake steps) is interrupted. If the previous step completes successfully, Y is authenticated, and the process continues with the next step.

As a result of the completion of all steps, the proposed blockchain-based authentication is successfully performed. In order to establish a secure DTLS tunnel, standard WebRTC session initiation steps are then continued.

VII. Performance Evaluation

Many abilities of the proposed solution are realized by providing access to the public blockchain network (Ethereum). Figure 6 shows the average completion times of these abilities on the same hardware and under similar network conditions. Measurements were taken multiple times for each function and averaged.

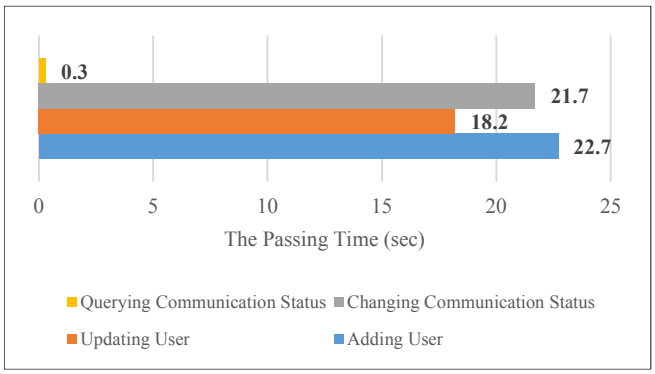

Fig. 6. Average completion times of transactions over the blockchain network

It is observed that methods that cause changes in blocks in blockchain when run, such as changing the user's(peer) authority to communicate, adding a new user, and updating the user's data can be completed in about 20 seconds. The length of these times is a result of the distributed structure of the public blockchain. Since the related methods are performed before real-time peer-to-peer communication is initiated, it does not affect the initiation (session negotiation-handshake) process of WebRTC communication.

When operated, the method which queries fingerprint and authority to communicate does not cause any change in blocks on the public blockchain. For this reason, it is quite fast compared to other functions mentioned in the previous paragraph. Since this method is implemented directly in the process of initiating real-time communication between peers, an unexpected delay in this function negatively affects the WebRTC session negotiation process.

Due to these comparisons, time-based evaluation can be applied between the standard WebRTC library and the proposed blockchain-based authentication solution. These comparisons were made on the same hardware under similar network conditions. The test results strengthened with five repetitions for both methods are shown in Figure 7.

As expected, the proposed secure mechanism was able to initiate WebRTC communication in a longer time than the standard library. The reason for this delay is the queries made through the public Ethereum network. It has been observed that this difference does not exceed 1 second for test environment conditions. This measured delay time; shows that

the cost of the proposed solution is within acceptable limits for real-life use-case scenarios.

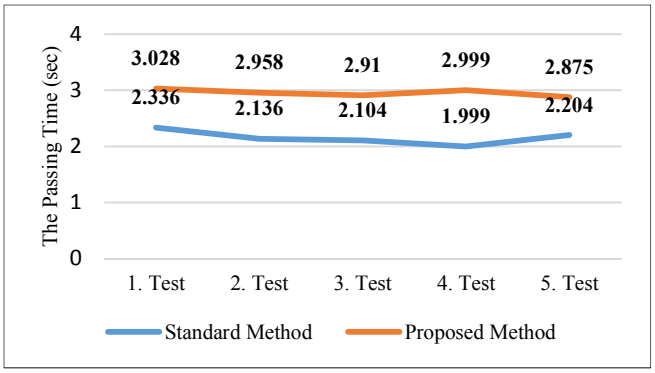

Fig. 7. Comparison of call setup times of the proposed method with the standard method

VIII. Conclusion

In communications using the standard WebRTC library, a secure communication channel is created between peers, but these peers cannot mutually authenticate each other. Even when using the DTLS-SRTP security approach proposed by WebRTC, there is no way to prevent man in the middle (MiTM) attacks. Some studies suggest various methods to solve this problem. For example, the developers of the WebRTC library propose an external identity provider mechanism for authentication. The problem here is that the service or services used as identity providers should also be trusted. For instance, when authenticating via Facebook Connect identity provider, it is necessary both to trust Facebook, which is a private institution, and to accept that the previously defined account on this system is not fake.

The proposed blockchain-based smart contract mechanism is used as an identity provider in this work. Furthermore, interactive parameters are added to this proposal so that users' authority to communicate can be changed in real-time. In the literature, no similar approach as a solution to these types of problems is proposed. It can be concluded that this proposed solution can be applied easily as a solution to similar problems in future studies. Designing the work to operate on distributed blockchain networks ensures that one-way trust is eliminated for centralized systems. Thus, the authentication service can be provided for a specific communication group without a centralized authority that needs to be trusted.

Digital identity is a hot topic in the blockchain world. The proposed solution on this paper enables users (peers) who are given digital identities to communicate securely on WebRTC using the existing smart contract mechanism on Ethereum blockchain networks. In future studies, instead of the Ethereum, Hyperledger Indy [29] can be chosen. Indy is becoming increasingly more widespread in the industy standing out with very successful projects on private blockchain networks.

Acknowledgment

This work was carried out with the support of Aselsan Inc. Furthermore, in some parts of the study, the computer laboratory of Gazi University has been used.

References

[1] A. Sergiienko, “WebRTC Blueprints,” Packt Publishing Ltd, 2014.

[2] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008.

[3] V. Buterin, “Ethereum white paper,” GitHub repository, 2013.

[4] E. Unsal and O. Kocaoğlu, “Blok zinciri teknolojisi: Kullanım alanları, açık noktaları ve gelecek beklentileri,” Avrupa Bilim ve Teknoloji Dergisi, 2018.

[5] C. Toğay and A. Levi, “WebRTC based augmented secure communication,” 24th Signal Processing and Communication Application Conference (SIU), IEEE, pp. 1621–1624, 2016.

[6] B. Feher, L. Sidi, A Shabtai and R. Puzis, “The security of WebRTC,” arXiv preprint arXiv:1601.00184, 2016.

[7] D. Boldt and S. Ebers, D. Boldt, and S. Ebers, “Security Mechanisms for Signaling in WebRTC-Based Peer-to-Peer Networks,” International Conference on Computational Science and Computational Intelligence (CSCI), pp. 19–24, 2017.

[8] H. Bostani and J. C.Grégoire, “Usable authentication systems for real time web-based audio/video communications,” 20th Conference on Innovations in Clouds, Internet and Networks (ICIN), IEEE, pp. 117–121, 2017.

[9] A. Reiter, and A. Marsalek, “WebRTC: your privacy is at risk,” In Proceedings of the Symposium on Applied Computing, pp. 664–669, 2017.

[10] E. Özgümüş, “The Use Of Blockchain In Digital Identity Management Systems For Kyc (Know Your Customer) Processes In Banks And Its Evaluation From Cybersecurity Perspective,” Master Thesis, Bahcesehir University Institute of Sşcebce, pp. 1–13 2018.

[11] A. Z. Ourad, B. Belgacem, and K. Salah, “Using blockchain for IOT access control and authentication management,” In International Conference on Internet of Things, pp. 150–164. Springer, Cham, 2018.

[12] R. Bagchi, “Using blockchain technology and smart contracts for access management in IoT devices,” Computer Science, 80, 2017.

[13] M. J. Werner, “WebRTC Security in the context of a DHT implementation,” http://www. inet. haw-hamburg. de/teaching/ss-2013/master-projekt/WernerMaxJonas_AW2. Pdf, 2013.

[14] M. Handley, V. Jacobson, and C. Perkins, “SDP: session description protocol,” 1998.

[15] F. Andreasen, M. Baugher and D. Wing, “Session description protocol (SDP) security descriptions for media streams,” RFC4568, 2006.

[16] D. McGrew, and E. Rescorla, “Datagram transport layer security (DTLS) extension to establish keys for the secure real-time transport protocol (SRTP)”, 2010.

[17] T. Dierks, and E. Rescorla, “The transport layer security (TLS) protocol version 1.2,” 2018.

[18] A. Bergkvist, D. C. Burnett, C. Jennings, A. Narayanan, and B. Aboba, “Webrtc 1.0: Real-time communication between browsers,” Working draft, W3C, 91. 2012.

[19] B. Feher, L. Sidi, A. Shabtai, and R. Puzis, “The security of WebRTC,” arXiv preprint arXiv:1601.00184, 2016.

[20] “Bitcoin Nedir?,” BtcTurk, https://pro.btcturk.com/faydali-bilgiler/bitcoin-nedir, Accessed 07-Feb-2020.

[21] T. Sert, “Sorularla Blockchain,” Blockchain Türkiye, Türkiye Bilişim Vakfı, 2019.

[22] “Ethereum.org,” https://www.ethereum.org/, Accessed: 08-Feb-2020.

[23] V. Buterin, “A next-generation smart contract and decentralized application platform,” white paper, 3(37), 2014.

[24] Ö. Engin, Coinkolik, “Ethereum Nedir? Yeni Başlayanlar için Ethereum Rehberi,” https://www.coinkolik.com/ethereum-nedir/, Accessed 08-Feb-2020.

[25] Ethereum Nedir?, Bitlo, https://www.bitlo.com/rehber/ethereum-nedir, Accessed: 08-Feb-2020.

[26] N. Szabo, Smart contracts. Unpublished manuscript, 1994.

[27] A. Usta and S. Doğantekin, “Blockchain 101 v2,” Bankalar Arası Kart Merkezi, 2017.

[28] Rinkeby Test Network, https://www.rinkeby.io/#stats, Accessed: 08-Feb-2020.

[29] Hyperledger Indy, https://www.hyperledger.org/projects/hyperledger-indy, Accessed 09-Feb-2020.