Ucam: A User-Centric, Blockchain-Based and End-to-End Secure Home IP Camera System

Xinxin Fan(✉), Zhi Zhong, Qi Chai, and Dong Guo

IoTeX , Menlo Park 94025, USA
{xinxin,zhi,raullen,dong}@iotex.io

Abstract. Home IP cameras are consistently among the most popular smart home devices and recent news stories about home IP cameras getting hacked frequently have posed serious security and privacy concerns for consumers. In this paper, we propose Ucam, a user-centric, blockchain-based and end-to-end secure home IP camera system. Ucam leverages advanced technologies such as blockchain, end-to-end encryption and trusted computing to address a number of vulnerabilities in the existing solutions. In the Ucam design, we replace traditional username/password based login approach with a one-click, blockchain-based passwordless counterpart and apply the resurrecting duckling security model to secure device binding. In particular, we utilize blockchain extensively to manage device ownership and provide integrity protection for the video clips stored locally or remotely. For coping with privacy, the end-to-end encryption, which is coupled with a user-centric, secure element enhanced key management scheme, is implemented in Ucam. Finally, Ucam employs re-encryption with Intel SGX as well as key refreshing to enable the sharing of encrypted video clips and live streaming videos, respectively. The security analysis and performance evaluation demonstrate that Ucam is able to meet the increasing security and privacy requirements for home IP camera systems with negligible performance overhead.

Keywords: Home IP Camera · Blockchain · Passwordless · End-to-end encryption · Integrity protection · Trusted computing

1 Introduction

The growing adoption of smart homes and expanding consciousness regarding security and safety have increased the demand for Internet Protocol (IP) based camera systems at a staggering rate. With packed features from face recognition to various image sensors and multiple connectivity options, home IP cameras offer a number of key benefits such as remote and perimeter video surveillance, intruder detection and alarms, access control and security management, etc.

N. Park et al. (Eds.): SecureComm 2020, LNICST 336, pp. 311–323, 2020.
https://doi.org/10.1007/978-3-030-63095-9_20

While home IP camera systems redefine safety and protection of properties and businesses, security and privacy of those systems continue to be major concerns for consumers [10]. Recent news [8,18,21] about hackers breaking into home IP camera systems has exposed a number of security design issues, including but not limited to poor password policies, problematical login process, vulnerable firmware and leaky database. Those vulnerabilities allow attackers to gain control of devices remotely and put users' personal information at risk.

First of all, the traditional password-based login has become the root cause for many recent hacks against home IP camera systems in which hackers launch so-called credential stuffing attacks [1] to access an account using a list of compromised login credentials. While these attacks could be mitigated by enabling the two-factor authentication mechanism [20], the complexity of the login process has been increased accordingly. Besides cumbersome login hurdles, the device binding mechanism that associates a user's account with his/her IP camera poses another major threat to the device ownership [3]. The third issue involves home IP camera systems that utilize cloud services for storing video clips, in which video files are stored either in plaintext or in encrypted form with the encryption key held by cloud service providers (CSPs) and/or device manufacturers. This practice exposes users' private information to third-party entities and allows them to manipulate the stored video clips in an arbitrary manner. Last but not least, while most home IP camera systems enable owners to share live camera feeds with friends and family, the video clips stored on the local SD card or cloud storage are only accessible by the camera owners. In particular, how to share encrypted video clips and live streaming videos has not been solved. To address the aforementioned issues for the existing home IP camera systems, we propose Ucam, a user-centric and end-to-end secure home IP camera system, in this contribution. Ucam leverages a blockchain wallet generated on the mobile app to enhance security of the user login process by realizing a one-click, password-less user authentication mechanism. Moreover, the resurrecting duckling security model [17] is applied to cameras in the system for securely binding devices with their owners. In particular, the critical device ownership information is directly anchored to the blockchain by cameras in lieu of being maintained by a centralized database server, which reduces the risk that cameras are taken over by attackers significantly. For protecting users' privacy, Ucam realizes end-to-end encryption coupled with a user-centric key management scheme. To thwart potential system errors and misbehavior of CSPs, Ucam allows cameras to periodically commit integrity checkpoints to the blockchain via user configuration, thereby enabling users to check data integrity when downloading video clips from the local or remote storage. Finally, Ucam realizes effective sharing of encrypted video clips and live streaming videos through re-encryption with Intel SGX and key refreshing, respectively.

The rest of the paper is organized as follows. Section 2 gives a brief overview of blockchain, smart contract, and Intel SGX. Section 3 describes the system and attacker models. Section 4 presents the detailed design of the Ucam system. In Sect. 5, we summarize the security and privacy properties of the Ucam design

and compare it with other home IP camera solutions. The performance impact of using a secure element in the Ucam system is evaluated in Sect. 6. Finally, we conclude this paper in Sect. 7.

2 Preliminaries

2.1 Blockchain and Smart Contract

Blockchains are tamper evident and tamper resistant digital ledgers implemented in a distributed fashion and usually without a central authority [23]. A blockchain is able to eliminate trusted intermediaries by requiring transactions to be verified by the rest of the blockchain's network. In particular, a distributed consensus protocol, which tolerates faults and adversarial attacks, ensures that all the nodes agree on a unique order in which blocks are appended. The blockchain provides an infrastructure where trust is embodied algorithmically in the transaction itself and effectively liberates data that was previously kept in safeguarded silos. In the context of blockchain, a smart contract [19] represents a piece of code that is stored, verified and executed on a blockchain. While the blockchain holds the storage file of a smart contract, a network of miners execute its business logic and update the blockchain by reaching a consensus. Users can invoke a smart contract by sending transactions to the contract address and each of them triggers the state transition of the contract, with data being written to the contract's internal storage. During the run-time, the smart contract performs predefined logic and may also interact with other accounts by sending messages or transferring funds. As self-executing codes on a blockchain, smart contracts are able to streamline processes that are currently spread across multiple parties and systems.

2.2 Intel SGX

The Intel Software Guard Extensions (SGX) [11] is a set of new x86 instructions provided in newer lines of Intel CPUs that allows application developers to protect sensitive data from unauthorized modification and access from rogue software running at higher privilege levels. SGX aims to provide a trusted execution environment (TEE) for user-space applications by enabling code isolation within virtual containers called enclaves. The program running inside an enclave is cryptographically measured and the generated proofs by the enclave can be reported back to the client. Enclaves feature three salient security properties, namely isolation, sealing and attestation [4]. *Isolation* means that program and data inside an enclave cannot be read/modified by other processes running at the same or higher privilege levels. On the other hand, *sealing* is the process of encrypting enclave secrets for persistent storage to disk, which uses authenticated encryption (i.e., AES-GCM) and thus allows the enclave to detect whether the sealed data has been modified externally. Finally, *attestation* enables an enclave to cryptographically prove that it is a genuine SGX enclave running on an up-to-date platform.

3 System and Attacker Models

3.1 System Model

We consider a blockchain-enabled home IP camera system as shown in Fig. 1, which consists of the following entities:

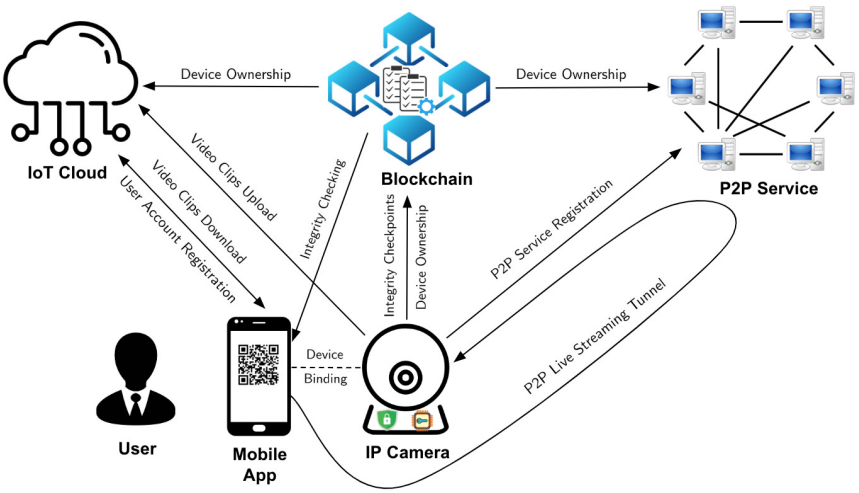

Fig. 1. The system model of a blockchain-enabled home IP camera system

- *IP camera*: An IP camera is a type of digital video camera which can receive control commands and send image data via the Internet.
- *IoT cloud*: An IoT cloud is responsible for user account management, device management, and data storage.
- *Peer-to-peer (P2P) service*: A P2P service simplifies the linkage between IP cameras and mobile devices when a user views camera feeds.
- *Mobile App*: A mobile app facilitates a smartphone user to configure an IP camera, view the captured video clips as well as live streaming videos, and share the video clips with friends and family.
- *User*: A user is the owner of one or multiple IP cameras and utilizes the mobile app to interact with them.
- *Blockchain*: A blockchain is a distributed ledger that is used to record transactions in the order agreed by all the peer computers in the network.

In the above home IP camera system, an IP camera, which is equipped with a secure element for key storage and cryptographic hardware acceleration, starts running once it is initialized and configured by a user via the mobile app. The IP camera records a short video clip (e.g., 10 s) and stores it either in the local storage (e.g., an SD card) or on the remote IoT cloud each time a motion is detected.

The user will receive an alarm and is able to replay the stored video clip using the mobile app. Moreover, the user can also request to view live streaming videos through the P2P streaming service in the system. The IoT cloud provides remote storage and serves users' requests for retrieving video clips. The blockchain, on the other hand, enforces device ownership, facilitates device sharing, and ensures data integrity of the cloud storage.

3.2 Attacker Model

In a typical home IP camera system, an adversary might try to compromise the user account system on the cloud server for taking over the ownership of IP cameras. A nearby adversary may also launch the attacks against the device binding process and take control of the victim's device. In addition, an attacker might eavesdrop on wireless communications between the IP camera and IoT cloud. We also consider the scenario in which a cloud provider could behave maliciously by viewing, inserting, deleting, and modifying the video clips. Furthermore, an attacker may impersonate a legitimate user and try to access the live streaming videos via the P2P service.

4 The Ucam Design

4.1 Passwordless User Authentication

To thwart potential credential stuffing attacks and improve user experience, the Ucam mobile app utilizes the private/public key pair associated with a user's blockchain wallet to implement passwordless login to the IoT cloud. In the Ucam system, the one-click, passwordless user authentication works as follows:

- Once a user opens the Ucam mobile app for the first time, a blockchain wallet is generated automatically, where the private key $priv_U$ is stored in the secure storage of his/her smartphone and the blockchain address $addr_U$, which is derived from the public key pub_U, is passed to the IoT cloud for user account creation. In the Ucam system, each user account consists of a blockchain address $addr_U$ and a random challenge r_U.
- When the user clicks on the login button, an API call to the IoT cloud is made for retrieving the random challenge r_U associated with the blockchain address $addr_U$.
- Upon receiving the random challenge r_U and displaying it on the Ucam mobile app, it requires the user's confirmation for the signed message r_U. If the user accepts it, a signature $\mathsf{Sign}_{priv_U}(r_U)$ is generated and returned to the IoT cloud together with the corresponding blockchain address $addr_U$. Otherwise, the login process is terminated.
- When the IoT cloud receives an authentication response, it first looks up the user account using the blockchain address $addr_U$ and obtains the current random challenge r_U, followed by verifying the authentication response

$\mathsf{Verify}(r_U, addr_U, \mathsf{Sign}_{priv_U}(r_U))$. If the verification succeeds, the user is considered as authenticated and a JSON Web Token (JWT) is issued to the user for accessing the cloud storage. Otherwise, the login attempt is rejected.
- The IoT cloud needs to update the random challenge r_U after each login attempt for thwarting replay attacks.

The above login process uses asymmetric cryptography and blockchain technology to eliminate the need of cumbersome passwords, thereby achieving better usability and security than the traditional username/password based approach.

4.2 Blockchain-Based Ownership Management

In the Ucam system, we bind a home IP camera with a user's account through the out-of-band (OOB) channel and apply the resurrecting duckling security model [17] in the context of IoT device binding. Once the camera is powered on for the first time or the reset button is pressed, the device will look for a valid blockchain address and recognize the device owner as the first entity that provides it. Therefore, when the user opens the Ucam mobile app and adds the camera to his/her account, he/she needs to hold the smartphone in front of the camera and allow his/her blockchain address, which is encoded as a QR code on the Ucam mobile app, to be scanned by the camera. Upon receiving the user's blockchain address $addr_U$, the camera will invoke $\mathsf{SC_{om}}$, an ownership management smart contract deployed by the camera manufacture on the blockchain, with parameters $addr_C$ and $addr_U$, when the internet connection becomes available. Here the camera claims its ownership by creating an association of its blockchain address (i.e., $addr_C$) with its owner's one (i.e., $addr_U$) on the blockchain. The following three cases might occur: i) If $\mathsf{SC_{om}}$ does not have any entry containing $addr_C$, a new entry $(addr_C, addr_U)$ will be created in $\mathsf{SC_{om}}$; ii) If $\mathsf{SC_{om}}$ has already included the same entry $(addr_C, addr_U)$, its means that the camera is reset by its current owner and $\mathsf{SC_{om}}$ does not need to update the state; iii) If $\mathsf{SC_{om}}$ has an entry $(addr_C, addr'_U)$ with $addr'_U \neq addr_U$, it implies the transfer of ownership (see Sect. 4.7) and the blockchain address $addr_U$ of the new owner will replace the previous one $addr'_U$ in $\mathsf{SC_{om}}$.

4.3 End-to-End Encryption and User-Centric Key Management

The Ucam system leverages end-to-end encryption to protect confidentiality of both video clips and live stream videos. The raw video data is encrypted using user-specified encryption keys with the aid of a hardware-based cryptographic engine (CE) inside the secure element, before it is stored locally on an SD card, remotely on the cloud storage, or sent to the P2P streaming service. Given a video frame v of l bits and a video encryption key k_V, the cryptographic engine encrypts the video frame as follows:

$$v' = v \oplus \mathsf{KSG}(l, \mathsf{CE}(k_V, IV)),$$

where $\mathsf{KSG}(\cdot)$ is a keystream generator using the underlying CE and the l-bit key stream is XORed with the video frame v to generate the corresponding ciphertext v'. Here CE can be instantiated using a stream cipher or a block cipher operating on the stream cipher mode [5] and IV is an initialization vector.

For enabling a user to update the video encryption key k_V in a secure manner, a Key-Encryption-Key (KEK) k_E is first derived from the user's private key $priv_U$ on the Ucam mobile app, i.e.,

$$k_E = \mathsf{KDF}(priv_U, OtherInput),$$

where KDF can be any standardized key-derivation function [2]. '$OtherInput$' might include a random salt (i.e., a byte string), the length of the derived key, and other context-specific data, depending on the choice of a key-derivation function. Note that k_E is derived immediately after the blockchain wallet is created on the Ucam mobile app and transported to the camera together with the user's blockchain address via the QR code. Upon receiving the KEK k_E, the camera stores it inside the secure element. Whenever a user would like to update the video encryption key k_V, he/she first generates a new key k'_V on the Ucam mobile app and then encrypts it with the KEK k_E, i.e.,

$$c = \mathsf{Enc}(k_E, k'_V).$$

The ciphertext c is then sent to the camera through the public channel and replaces the previous encryption key in the file system. As a result, the subsequent video clips or live stream videos will be encrypted with the new key k'_V. Here we utilize two different keys k_C and k_S to encrypt video clips and live streaming videos, respectively, for accommodating the corresponding video sharing mechanisms (see Sects. 4.5 and 4.6) and k_V can be either of those keys.

4.4 Blockchain-Based Data Integrity Protection

For ensuring data integrity of video clips stored locally on an SD card or remotely on the cloud storage, the Ucam system allows the camera to commit integrity checkpoints to the blockchain according to a user-defined time period. To this end, the user needs to first enable the data integrity protection feature on the Ucam mobile app and specify the time period in days for checkpoint commitments, followed by topping up the camera's wallet with a certain amount of cryptocurrency tokens. Note that the shorter the time period is set, the more checkpoints the camera is going to commit on the blockchain.

Once the data integrity protection feature is activated, the camera starts building a Merkle tree [12] dynamically for the encrypted video clips received during the user-specified time period. At the end of each time period, the camera will invoke another manufacture deployed smart contract $\mathsf{SC}_{\mathsf{cm}}$, which is responsible for checkpoint management, with parameters (id_{mt}, num, h_r), where id_{mt} is the Merkle tree identifier that is concatenated with a file identifier to indicate which Merkle tree the file belongs to. h_r is the root of the Merkle tree built from num encrypted video files and acts as the integrity checkpoint for the past time

period. As soon as integrity checkpoints become available on the blockchain, the user is able to verify data integrity of encrypted video clips retrieved from the SD card or cloud storage. After the user sends a request for downloading an encrypted video clip from the SD card or cloud storage, the camera or cloud server first identifies all the encrypted video clips that are in the same Merkle tree as the one in question using the Merkle tree identifier id_{mt}, followed by the generation of the corresponding Merkle path. The encrypted video clip and Merkle path are then returned to the Ucam mobile app. Before decrypting the video clip, the Ucam mobile app obtains h_r from the smart contract $\mathsf{SC}_{\mathsf{cm}}$ and verifies data integrity of the received video clip using h_r and the Merkle path. In this way, the user is confident that the video clip has not been altered.

4.5 Fine-Grained Secure Video Clip Sharing with Intel SGX

Considering the limited video sharing scenarios of home IP cameras, we describe a fine-grained secure video clip sharing scheme through a re-encryption process using the Intel SGX technology. More specifically, we create a data sharing enclave DataShare on the application server of the IoT cloud, which is responsible for re-encrypting the video clip(s) selected by the user for sharing purposes. The video clip sharing process works as follows:

- Whenever a user wants to share the video clip(s) with others, the Ucam mobile app will first perform a remote attestation with the DataShare enclave to verify that the application server has loaded the correct code into the enclave. During this process, a symmetric session key k_{se} is generated on both the Ucam mobile app and DataShare enclave, thereby establishing a secure channel between two entities.
- After the user selects n video clip(s) on the Ucam mobile app and sets a video sharing key k_{sh}, the list of video file identifiers $\{id_{f_1}, \ldots, id_{f_n}\}$, the video sharing key k_{sh}, and the video clip encryption key k_C are encrypted using the session key k_{se} and sent to the application server.
- The application server sends the received information to the DataShare enclave for decryption and retrieves n encrypted video clip(s) from the cloud storage using the identifier list $\{id_{f_1}, \ldots, id_{f_n}\}$. The retrieved n video clip(s) are then decrypted and re-encrypted inside the DataShare enclave using k_C and k_{sh}, respectively.
- The application server stores the re-encrypted video clip(s) in the cloud and returns the Uniform Resource Identifier (URI) to the Ucam mobile app. The user is then able to share the URI and video sharing key k_{sh} with others via various communication channels (e.g., QR code, email, etc.).

To save costs for using cloud storage, the URI for the shared video clip(s) is only valid for a user-defined amount of time and all the shared video clip(s) will be deleted thereafter. The above secure data sharing scheme enables a user to fully control which video clip(s) to share with different entities, thereby minimizing the potential data leakage.

4.6 Secure Live Streaming Video Sharing with Key Refreshing

Due to the real-time requirements for sharing live streaming videos, we employ key refreshing in lieu of re-encryption for sharing cameras with other people. More specifically, the device owner will directly send the current live streaming video encryption key k_S to all the entities with which he/she would like to share the camera. Whenever the device owner decides to revoke access for one or multiple people, a new live streaming video encryption key k'_S will be generated on the fly and distributed to the remaining entities. Moreover, the device owner also needs to update the live streaming video encryption key on the camera as described in Sect. 4.3.

Besides distributing the live streaming video encryption key k_S, the device owner also needs to generate access tokens for authorizing other entities to retrieve live streaming videos via the P2P service. An access token TK_{OR} is a tuple $(pub_O, addr_R, addr_C, T_{exp}, \mathsf{Sign}_{priv_O}(addr_R, addr_C, T_{exp}))$, where pub_O is the device owner's public key. $addr_R$ and $addr_C$ denote the blockchain addresses of the requester and the owner's camera, respectively. T_{exp} is the expiry time of the access token and $\mathsf{Sign}_{priv_O}(addr_R, addr_C, T_{exp})$ is the device owner's signature. A data requester can retrieve live streaming videos with the access token as described below:

- The requester sends a connection request to the P2P service by presenting his/her public key pub_R and access token TK_{OR}.
- The P2P service verifies the validity of the access token TK_{OR} as follows:
  - The P2P service checks T_{exp} to ensure that the access toke TK_{OR} is not expired;
  - The P2P service queries the ownership management smart contract $\mathsf{SC}_{\mathsf{om}}$ with the device's blockchain address $addr_C$ and then obtains its owner's blockchain address $addr_O$;
  - The P2P service checks that $addr_O$ and $addr_R$ are derived from the public keys pub_O and pub_R, respectively;
  - The P2P service verifies that the signature $\mathsf{Sign}_{priv_O}(addr_R, addr_C, T_{exp})$ is valid.

  If any of the above verification steps fails, the P2P service will reject the connection request.
- The P2P service sends a random challenge r_P to the requester for verifying that he/she is the owner of the blockchain address $addr_R$.
- The requester generates the signature $\mathsf{Sign}_{priv_R}(r_P)$ and sends it to the P2P service as the response.
- The P2P service verifies the validity of the signature $\mathsf{Sign}_{priv_R}(r_P)$ and then grants or rejects the P2P service from the requester accordingly.

4.7 Ownership Transfer

Thanks to the ownership management with the smart contract on the blockchain, the ownership transfer can be easily handled. In the case that the camera is given to another person, the new owner can simply reset the camera, register a user account and restart the ownership claim process (see Sects. 4.1 and 4.2).

5 Security and Privacy Properties

Table 1 presents a comprehensive comparison of Ucam and other popular home IP camera systems in terms of security and privacy properties. Among the existing solutions, Ucam is the only one that utilizes secure hardware for protecting cryptographic keys and blockchain wallet for passwordless user authentication, respectively. While Haicam [7] and Wyze [22] claim the usage of end-to-end encryption, it is not clear how this technology is actually implemented and, in particular, how the encryption key is managed in their systems, due to limited technical information available on their websites. Regarding device ownership management, Ucam takes advantage of the decentralized nature of blockchain to achieve stronger protection of device ownership, when compared to other home IP camera systems in which centralized cloud servers are used for this purpose. Furthermore, Ucam offers additional integrity protection for video clips against storage errors and malicious attacks with the help of integrity checkpoints stored on the blockchain. As for video sharing, Ucam supports fine-grained sharing of encrypted video clips as well as secure sharing of live streaming videos. Although Wyze [22], eufy [6], Ring [15] and Nest [13] also implement (partial) video sharing functionalities, CSPs are still able to access the shared contents. From Table 1, we can see that Ucam provides a number of salient features that improve security and privacy of the state-of-the-art home IP camera systems dramatically.

Table 1. Security and Privacy Properties of Home IP Camera Systems

	Ucam	Haicam [7]	Wyze [22]	eufy [6]	Ring [15]	Nest [13]
Secure Hardware	✓	✗	✗	✗	✗	✗
User Login	Blockchain Wallet	Username/ Password	Username/ Password with 2FA	Username/ Password	Username/ Password with 2FA	Username/ Password with 2FA
End-to-End Encryption	✓	✓	✓	✗	✗	✗
User-Centric Key Management	✓	Unknown	Unknown	N/A	N/A	N/A
Device Ownership Management	Blockchain	Cloud Server				
Data Integrity Protection	✓ (Blockchain)	✗	✗	✗	✗	✗
(Encrypted) Live Streaming Video Sharing	✓ (Key Refreshing)	✗	✓	✓	✓	✓
(Encrypted) Video Clip Sharing	✓ (Re-Encryption)	✗	✗	✓	✓	✓

6 Performance Evaluation

In the Ucam design, a secure element serves as the secure key storage and hardware cryptographic accelerator. More specifically, the secure element is respon-

sible for signing transactions to secure device ownership and commit integrity checkpoints (see Sects. 4.2 and 4.4) as well as generating keystreams in the end-to-end encryption (see Sect. 4.3), respectively. In this section, we use the EdgeLock^{TM} SE050 secure element development board [14] from NXP semiconductors to evaluate the performance impact. In a typical setting, a host controller communicates with an SE050 secure element through an I^2C (Inter-Integrated Circuit) interface. At the application level, the host controller exchanges messages with the SE050 secure element using application protocol data units (APDUs) [9]. To simplify the software development, NXP abstracts the interactions between the host controller with the SE050 secure element through the Plug & Trust middleware and associated secure sub-system (SSS) APIs.

For implementing the blockchain-based ownership management and data integrity protection protocols in the Ucam design, the SE050 secure element is used to sign transactions that are sent from the host controller. We configure the SE050 secure element to use the Koblitz curve `secp256k1` [16] and test the performance of ECDSA on a message digest of 32 bytes. Our experimental result shows that a digital signature can be generated in around 45.4 ms. In regard to the end-to-end encryption in the Ucam system, we use AES-128 in the counter mode (CTR) [5] to generate keystreams for encrypting video frames. For testing the throughput of the keystream generation on the SE050 secure element, the host controller sends 16-byte messages, each of which consists of a 12-byte nonce and a 4-byte counter, to the AES engine consecutively. The resulting keystream generation throughput is about 11.3 Kbps. Note that all the performance test results take the I^2C communication between the host controller and the SE050 secure element into consideration. Based on our experimental results, one can see that secure elements are able to boost security of home IP camera systems dramatically without incurring significant performance overhead.

7 Conclusion

In this paper, we present the design of Ucam, a user-centric, blockchain-based and end-to-end secure home IP camera system. When compared to popular home IP camera solutions, Ucam offers strong security and privacy protection for multiple core functionalities such as user login, device binding, device ownership management, video confidentiality, storage integrity and video sharing. By leveraging blockchain technology, Ucam is able to support passwordless user authentication and protect cameras and videos from various malicious attacks. Moreover, the video data is only accessible by the camera owner and their authorized entities, thanks to the end-to-end encryption and user-centric key management. The secure sharing of encrypted video clips and live streaming videos is addressed using re-encryption based on Intel SGX and key refreshing techniques.

References

1. Avast Security News Team, "What is credential stuffing, and why is my smart security camera vulnerable to it?". https://blog.avast.com/credential-stuffing-and-web-cams, Security News, 4 May 2019
2. Chen, L.: Recommendation for Key Derivation Using Pseudorandom Functions (Revised), NIST Special Publication 800–108, October 2009
3. Chen, J., Sun, M., Zhang, K.: Security analysis of device binding for IP-based IoT devices. In: Proceedings of 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), IEEE Computer Society, pp. 900–905 (2019)
4. Costan, V., Devadas, S.: "Intel SGX Explained", IACR Cryptology ePrint Archive, Report 2016/86 (2016)
5. Dworkin, M.: "Recommendation for Block Cipher Modes of Operation: Methods and Techniques", National Institute of Standards and Technology, NIST Special Publication 800–38A, December 2001
6. eufy Security Indoor Cam 2K Pan & Tilt. https://www.eufylife.com/products/variant/eufycam-2/T8410121
7. Haicam End-to-End Encrypted Home Security Camera. https://haicam.tech/
8. Isidore, C.: Smart camera maker Wyze hit with customer data breach. https://www.cnn.com/2019/12/30/tech/wyze-data-breach/index.html, CNN Business, 30 December 2019
9. ISO/IEC 7816–4:2013, Identification cards - Integrated circuit cards - Part 4: Organization, security and commands for interchange
10. McInnis, K.: Consumer Reports letter to connected camera manufacturers to call for raising security and privacy standards. https://advocacy.consumerreports.org/research/consumer-reports-letter-to-connected-camera-manufacturers-to-call-for-raising-security-and-privacy-standards/, Consumer Reports, 13 January 2020
11. McKeen, F., et al.: Innovative instructions and software model for isolated execution. In: Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy (HASP 2013), p. 10. ACM Press (2013)
12. Merkle, Ralph C.: A digital signature based on a conventional encryption function. In: Pomerance, Carl (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 369–378. Springer, Heidelberg (1988). https://doi.org/10.1007/3-540-48184-2_32
13. Nest Cam Indoor. https://store.google.com/us/product/nest_cam
14. NXP Semiconductors. EdgeLock^{TM} SE050 Development Kit
15. Ring Indoor Cam. https://shop.ring.com/products/mini-indoor-security-camera?variant=30258040832089
16. Standards for Efficient Cryptography. "SEC 2: Recommended Elliptic Curve Domain Parameters, Version 2.0", Certicom Research (2010)
17. Stajano, F., Anderson, R.: The resurrecting duckling: security issues for ubiquitous computing. Computer **35**, 22–26 (2002). IEEE Computer Society
18. Sundby, A.: Hacker spoke to baby, hurled obscenities at couple using Nest camera, dad says. https://www.cbsnews.com/news/nest-camera-hacked-hacker-spoke-to-baby-hurled-obscenities-at-couple-using-nest-camera-dad-says/, CBS News, 31 January 2019
19. Szabo, N.: Smart Contracts: Building Blocks for Digital Markets (1996). http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html

20. Thomas, K., Moscicki, A.: New research: how effective is basic account hygiene at preventing hijacking, Google Security Blog, 17 May 2019
21. Vigdor, N.: Somebody's Watching: Hackers Breach Ring Home Security Cameras. https://www.nytimes.com/2019/12/15/us/Hacked-ring-home-security-cameras.html, The New York Times, 15 December 2019
22. Wyze Cam V2. https://wyze.com/wyze-cam.html
23. Yaga, D., Mell, P., Roby, N., Scarfone, K.: Blockchain Technology Overview, National Institute of Standards and Technology, Draft NISTIR 8202, January 2018