Hardware-rooted KAC for scalable data sharing in cloud

Chandu Komme <c.komme@stud.fh-sm.de>, 319570

1 Abstract

This project presents a hardware-rooted, device-bound data-sharing system on Raspberry Pi that ensures *where* decryption happens, *who* is physically present, and *which* data classes a device may access-while keeping plaintext strictly at the edge. The main aspect of the design is a Key-Aggregate Cryptosystem (KAC) that condenses decryption rights to any specified group of classes into one constant-size aggregate key that the device is provided with. Together with per-file AES-GCM using a new 256-bit key and a 96-bit nonce, the system not only eliminates per-file key distribution, it also supports instant changes to policy through the rotation of a single key and avoids exposing the plaintext even in the event of a cloud attack. Each object has a very small header of the needed object decryptor information, a name of the class, nonce, authentication tag, KAC wrapping of the file key, integrity digest and optional environment snapshot attached to each object AAD, and binds the policy context, including classes and key ID in order to cause tag failure on header corruption. The process of decryption is the presence-gated one: prior to any of the operations, there must be the MFRC522 RFID tap. It is displayed using an SSD1306 OLED with explicit states (Ready, Tap, Granted, Denied, Integrity Failure), a secure element (conn.) can be selected that signs a fresh challenge with a different providing a direct route to a server-verified liveness awareness; a sensor (BME280) may be provided that will provide temperature, humidity and pressure as provenance backgrounds. Role cloud The role plays with a deliberately simple flask application that stores ciphertext bytes and the header of the package, but never tries to process plaintext bytes or long term keys. The Pi is coded with the encryption and de-encryption so that the large files can be processed efficiently with only mistakes which are deterministic and explicit so that the auditing capability can be achievable the small header with its privacy focused is mathematically defined in terms of the interoperability and it is defined in words where the wiring and software modules are post hoc on the Pi based on I 2 C OLED BME280 ATECC608 elle LED Query. In the qualitative results, interactive tap-to-decrypt latency across local networks; a capability to deny packets with incorrect format, error in classes and an ability to alter an encrypted ciphertext with ease; policy agility with ease through simple replacement of a single aggregate key, rather than

reencryption of stored data exhibit itself. The design is resistant to a damaged cloud because the server is never provided with plain texts or device secrets, and it is never challenging to switch over to dual wraps key rotation. Overall, the system provides an effective, repeatable framework to the small teams and IoT-specific environments that require an authorization framework based on classes, the device conveyed decryption, and the lean and audit-able cloud presence. Future retention includes server-authenticated default attestation, less aggressive header compression on the mandatory TLS, Scale Key Information Life-cycle management, and that of hybriding migration to post-quantum KEMs and signature, and to retain the existing roles and workflows.

Keywords: Key–Aggregate Cryptosystem (KAC), AES-GCM, Raspberry Pi 4B, MFRC522 RFID, ATECC608A, SSD1306 OLED, Flask, Edge Security, Device-Bound Decryption, BME280, Class-Scoped Authorization, Attestation, Revocation, Privacy-Preserving Cloud.

2 Introduction

An example of the classical cloud-based collaboration system is when it is assumed that only the key-holders have an opportunity to obtain access to the decryption. However, the key duplication, SD card cloning and access to a server dissociates the authorization with the device as well as the human presence. The suggested system will comprise Raspberry Pi 4B and Key-Aggregate Cryptosystem (KAC) to specify authorization using the fixed-size key in order to allow all the targeted classes of messages to be authorized using only one key. This will eliminate the necessity of key exchange on every file, and make live change of policy by merely rotating one key.

There are three subdivisions in the architecture. Manager is also used to make aggregate keys with Setup and KeyGen/Extract. It has the Encrypt/Decrypting AES-GCM of fresh keys and nonces at the Member (Raspberry Pi) to generate ciphertext and a small JavaScript Object (class, nonce, tag, KAC wrap, integrity digests and optional context). Cloud is flask server which is lightweight and provides support of uploading and downloading items. Its processing is not more than ciphertext. No long-term keys are stored.

An RFID MFRC522 tap decrypts and intentionally to an SSD1306 OLED display decrypt on the operator side. Signing Hardware liveliness may be signed assisted by optional (signing) ATECC608A secure element and optional (tracking) BME280 capability tracking may be provided. The KAC Set up, KeyGen/Extract, Encrypt, Decrypt step of the operations, assisted by the application of the 3.3 V I 2 C / SPI connection, directly proportionately relates to the code construction requirements of the code.

3 Literature review

Granting small groups of users and IoT applications a cloud based protection has longs been a trade-off of two goals: flexible authorization of a wide variety of documents or streams, and operational simplicity on limited edge devices. In the initial designs, a separate symmetric key was assigned to each file or folder and this complicated distribution

and it was also costly to rotate. Attribute-based designs shifted policy to cryptography, and tended to increase key and ciphertext size as policy complexity increased, becoming a barrier to edges adoption [2][7]. Key-Aggregate Cryptosystems solve this by a compression of the decryption privileges of any selected set of classes into a fixed-size aggregate key. A device that has been granted multiple classes has a single key and can be rotated or revoked very quickly without mobiles undergoing a significant re-encryption change, or even through a semi-trusted proxy involved [1].

The new standard of domain of confidentiality and integrity at rest and in transit as well as at the object layer with AES-GCM focus appears to be authenticated encryption with associated data. One of the literature focuses to enable nonce memory budgets includes nonce uniqueness, nonce binding policy context and nonce streaming encryptioning of large files. In this implementation, both aggregate key and AEAD tag give response to either the possibility of a device to acquire a per-file key at a certain step or not and integrity in the case of obtaining a key respectively, which causes any type of tampering on the header or ciphertext to be closed to fail.[10][6][1].

An analogous thread is concerned with trust that is bound on devices. Long term credentials which are external to the filesystem are then safeguarded by secure elements and trusted modules and may sign per request challenges to provide evidence of liveness. The downloads are bound to a challenge which has been signed by a registered device key, lessening the risk of exfiltrating the key and allowing it to be rate limited and audited, without ever touching user plaintext itself off of the device[3][4]. At small single-board computer sizes, I moved around components based on I2C ic secure elements are considered by dint of the easily wiring nature, or low energy consumption, but supply-chain assurance and bus exposure continue to be realistic issues. The mitigations that have been recommended are authenticated bootstrapping, key- attestation logs, transport-layer protection that would hide metadata [3][9].

Presence factors including RFID provide a low friction human-in-the-loop gate. Previous literature considers low-cost RFID as a presence signal, but not a robust identity authentication, and by adding cryptographic authorization to the device, a convenient scheme can be adopted to prevent unattended decrypts by a simple session of tapping and going[5]. Arguing in favor of visible cryptography via is also human-factors research pointing out the importance of configurable small-screen status using concise formatting on a small OLED display and clear pass-fail LEDs to decode abstract states into readable form and minimize operator error in time-sensitive activities [6].

Minimalism is a common design objective as seen in the cloud-service viewpoint. End-to-end encrypted storage does propose dumb object stores, tokenizing bytes and compact header keys but never decrypting, and never continuously user key, to have a smaller impact of breach and compliance footprint. Best practices are recommended to be randomized names of objects, severe separation of ciphertext and metadata, omission of plaintext digests in the visible network headers, and basic upload and download micro-frameworks. among the cutover operations operationally smooth rotation is possible through supporting dual wraps: new and old aggregates coexist during a short time without relocating data [1][7][9].

More and more edge security provenance and context are discussed. Objects that have

been encrypted may be annotated with snapshots, encompassing temperature or humidity, to facilitate audit and forensics by similar environmental sensors and may enable this data to be regarded as non-sensitive or secured during transport and, where required, captured in correlative data to ensure that a discrepancy is not found. Higher voltages Extended literature on the IoT also focuses on tuned voltage-level matching, crude bus diagnostics and Mozilla energy and latency performance integrating streaming cryptography on the single-board system [6].

In future developments, post-quantum migration is suggested to be guided by the recommendation of hybridization between existing designs by substituting the use of public-key wraps and public-key signatures with standardized post-quantum mechanisms, without changing the roles and flows. Since, with a symmetric layer, class-scoped compression is not dependent upon the symmetric layer itself, the performance of aggregate keys, rapid rotation, and a plaintext-blind cloud can be largely transferred; just the artifact sizes and compute expenses are altered. Hybrid header may be based on transitional deployments until fleet upgrades are done [8].

Combined, the previous literature brings together four themes informing this project, namely, aggregate keys to offer compact, class-scoped device-authorization [1]; AEAD to offer per-object confidentiality as well as integrity along with policy-constrained associated data [10], device-bound trust by offering secure-element liveness and human-readable presence gates [3][5] and a minimalistic, plaintext-blind cloud with meticulously maintained metadata hygiene and operationally simple rotation [7][9][6]. The resultant architecture is both scholarly and viable to limited resource edge hardware.

4 Scientific Background and Up-to-Date Knowledge

The current trend in key sharing is towards edge-first encryption in the implementation of AEAD instances of device encryption, and sent encrypted data to non-trusted data storage only, such that the loss of a server does not disclose a plaintext. Key Authorization Key Authorization of individual sized aggregate keys The authorization is coded down such that rotation and revocation are both computationally inexpensive without necessarily involving huge re-encryption. Assurance means, type Assurance: Assured liveness Assurance is provided by assurance device-bound assurance together with crypto: Low friction RFID provides the human presence gating and secure element (e.g., ATECC608A) sign challenge to prove liveness. Best practice links policy context (a class, a key ID) to GCM related data and copies large files in order to be able to fit edge resources. Majority of recent instructions also aim at performing post-quantum specific transition to replace public-key wraps and signatures but does not replace roles and functions.

5 Selected Open Topic: Future of Raspberry Pi for Device-Bound, Class-Based Cloud Sharing

Raspberry Pi also can serve adequately as a prototype host platform to advance to a full-fledged edge anchor that can be common to classes and connected to devices. Future

development possibilities are parity on default (attestation on default or reversal of all download default challenges), and or measured boot or anti-rollback with policy-as-code (approving classes, lifetime and rotation rate). Presence could become optional PINs, more traditional RFID to touch unlocks, and headers consistent are skinny and subject to bound corresponding data and TLS, which must be provided. It is presumed that the opaque configuration relay helps in the operations of the fleet, and telemetry is rolled in phases during the cutover operations and privacy-preserving configuration generates results without revealing identity or content.In case the post-quantum KEMS and signatures applied on the mid-term hybrid headers do not breach the platform, then no roles and tasks are lost, and the platform continues to operate in the cryptographic duties. The side-channel and bus sniffing threats may be overcome using the protected I 2 C, epoxy or conformal module coatings, rate-limited download APIs and keyed audit logs, which are bound to counters on the secure element. With the development of a good OLED user interface, and user friendly, then pagination of long text should be feasible, it should be able to give traces of provenance (BME280 snapshots), and reflect failure, which can be analyzed.

6 Methodology

This project offers a sharing by classes through Raspberry Pi which has a ground in the equipment and is confined by a machine. It applies internal validity to performability and hardness by seeking to cope with full Key-Aggregate Cryptosystem (KAC)-life cycle (Setup, KeyGen/Extract, Encrypt, Decrypt) with physical presence controlled gating, non-compliant optional secure-element liveness and weak cloud.

6.1 Environment Setup

- **Roles and topology:** Member (On-device encrypt/decrypt) Raspberry Pi 4B. Cloud (store ciphertext and headers) is a light service which has been migrated to Cloud. Manager-executable scripts are executed by Pi or any other Linux machine to pass parameters and total keys to favored classes (finance, iot, etc.).
- **Pi hardwar(logic 3.3V):** MFRC522 RFID interface through SPI, MFRC522 which comprises: SSD1306, OLED display at I2C 0x3C, LEDs through GPIO17/27 using 330 Ohm resistors, optional ATECC608A at I2C 0x60, optional BME280 at 0x76/0x77
- **System setup (high level):**Eliminate multiplexing; introduce requirement of I 2; introduce dependencies of pycryptodomex, mfrc522, luma.oled, smbus2, and optional adafruit-circuitpython-bme280, Flask. Add a collective key of the device.

6.2 Security Feature Baseline

- **Edge-first AEAD:** Encoding files with AES-GCM is done in a new 256 bit key and 96 bit nonce alongside associated data containing the policy context (class, optional

kid) to ensure a failure of Tamper Header Attacks closed.

- **Class-scoped Authorization (KAC):** There is one fixed-sizeAggregate Keys in the Pi, which authorizes only the class-subsetof the Pi, with one-file turn-over, and not mass-reencrypt.
- **Presence and liveness:** An RFID tap is mandatory prior to any decrypt. Optionally, the ATECC608A signs a challenge as hardware liveness.
- **Header minimalism:** The Cloud holds ciphertext and compact JSON headers (class, nonce, tag, KAC wrap, integrity digests, optional environment). Plain text and long-term keys do not leave the Pi.

6.3 A cryptographic Elements and Implementation

- **Manager (Setup/KeyGen Extract):** Send system parameters and on an arbitrary selection of classes, the issue of an aggregate key to this device which is a device key; export a kid label to trace rotation.
- **Member -Encrypt:** A new AES key and nonce should be created, AES-GCM streaming needs to be achieved, ct-sha256 control (or optional pt256 control) needs to be achieved through the use of KAC file and finally file key encryption should be achieved using KAC file which should be then uploaded to the Cloud.
- **Member- Decrypt (presence gated)** RFID enroll tap windows; verifies header and cipher text, class is authorized, decrypt file key KAC-unwraps, verifies GCM tag, on success, plain text written, displayed on OLED (pager w/ UTF-8).

6.4 Simulation and Testing

Simulation was used to test and verify both correctness and operations. An approved group, working along with a registered RFID, still made a successful decryption in surgical controlled trials. No unwrapping key of unknown RFID or unauthorised class was permitted. AES-GCM tags malfunctions all leave plaintexts undeciphered due to spoilt headers or ciphers. On large inputs, streaming was applied to ensure that the memory was kept to a limit. At cutover files in different classes would be encrypted and uploaded as operational runs and several wraps would be key rolled over. Revocation instantly changed the operation of the gadget to reencrypt byzantining stored information to guarantee unhindered rotation as opposed to reencrypting.

6.5 Framework Alignment

Lifecycle mapping: KAC builds its own setup, KeyGen/Extract, Encrypt and Decrypt using a variety of hardware as well as software elements with human input in both presence gating and optional liveness wrap decrypt. The rotation is carried out by way of a child rotation while privacy hygiene includes the minimization of headers and use of optional TLS.

7 Implementation of device-bound, class-based sharing in the Real World

The next stage is our on-the-job implementation of our Raspberry Pi-based system that implements decryption of devices (which are device-bound) and authorization (which is device-based in its scope). Use of cloud-based documents by many small teams also share devices and credentials despite having them in the cloud. The ciphertext is also stored on the cloud along with a concise header, none of which defines plaintexts except in the Pi. Only the authorized classes have a single aggregate key and a user can do a quick RFID tap before any decrypt.

Scenario-based approach: Raspberry Pi serves as the representative of an XYZ company finance desk (Member Authenticated). Reports on various departments (e.g., finance, hr, iot) are uploaded by the staff. The Pi is given an aggregate key by the Manager with a permission to access finance only. All the files are encrypted together in the Pi using AES-GCM, and uploaded using the minimal Flask cloud using a small header (class, nonce, tag, KAC wrap, checksums, optional environment snapshot). A user taps an enrolled RFID card when he or she wishes to read a finance report. Provided that the class is granted and the integrity checks are successful, the Pi will decrypt and show the text in the OLED; other classes are immediately denied. Hardware liveness in downloads to servers Option ATECC608A requires signature verification of downloads.

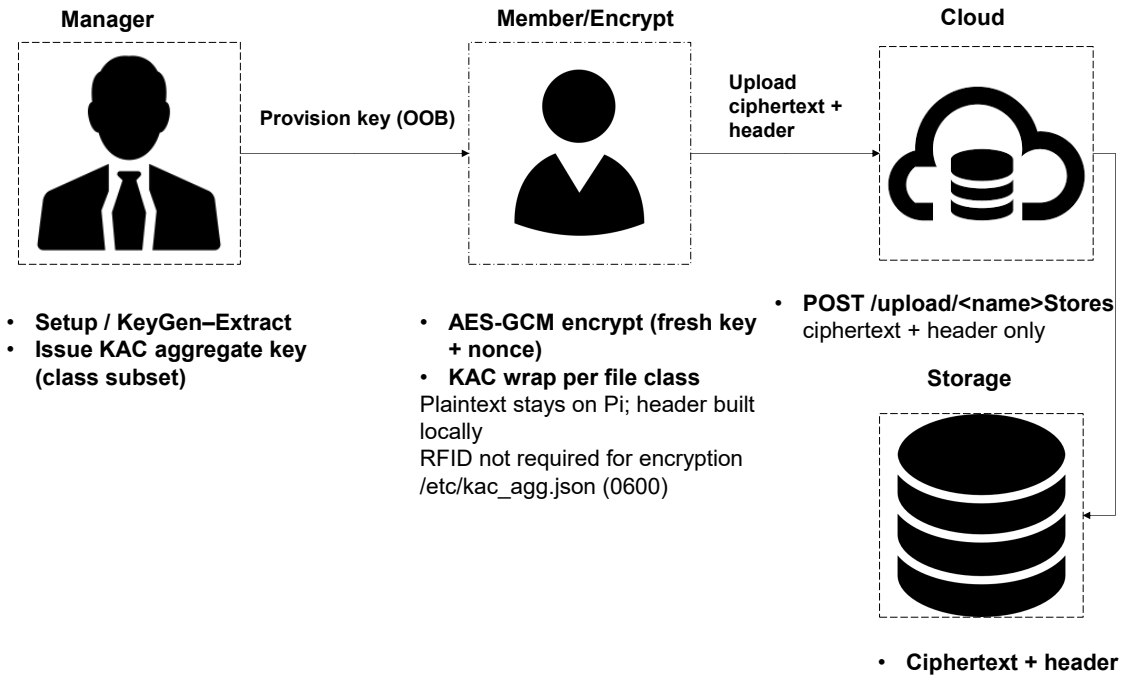

Figure 1: Member (Encrypt) — local AES-GCM + KAC wrap and upload.

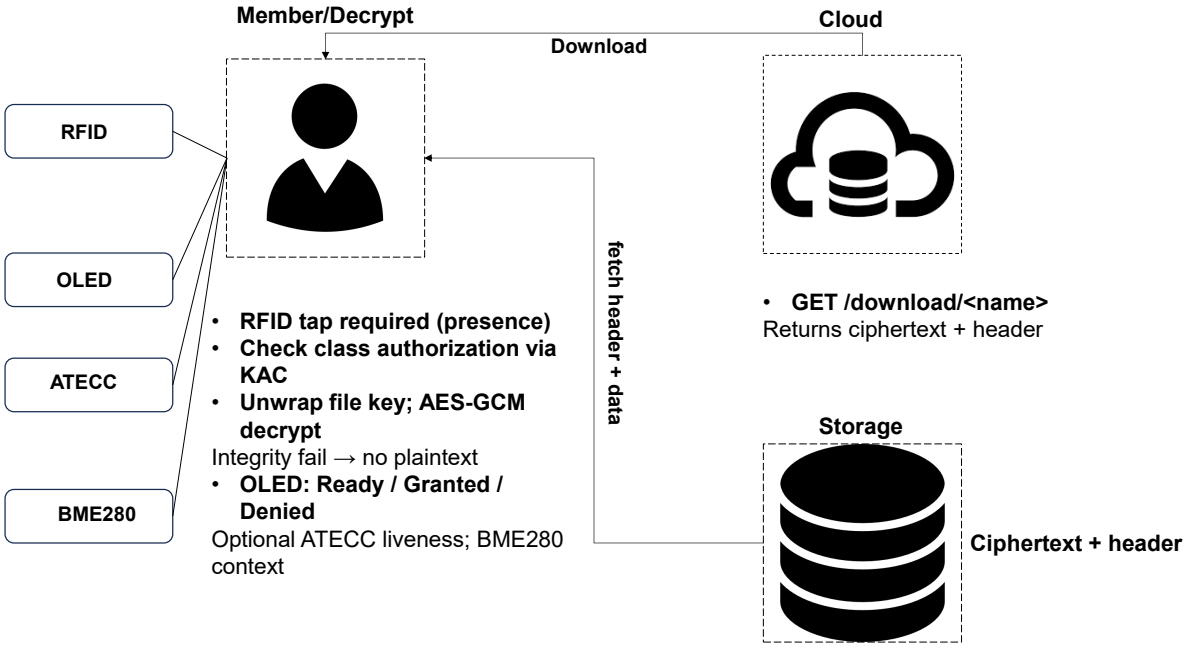

Figure 2: Member (Decrypt) — RFID-gated KAC unwrap and AES-GCM decrypt.

7.1 Member authorization and Walkthrough of OLED

Booting the OLED displays, the Ready-Tap card. An RFID tap up is downloaded as valid and enrolled with the downloading of ciphertext and header, which checks against the class authorization, KAC unwrap and checked using AES-GCM. Further, it displays Access Granted-Name and renders the text by OLED, paged. The areas that have not been set and do not bear any RFID identifier say Access Denied. Any alteration of the header or ciphertext produces Integrity Failure and there is no plaintext.

7.2 Python encrypt, uploading and decrypt python scripts

- **Encryption is run locally:** usage An example of encrypting a file with Python is as follows: python3 member/encrypt.py file -t - Encrypts file with Python, calculates checksums and requires file to create a KAC wrap on file and uploads final ciphertext.
- **Software-Secured transmit at the sender side:** python3 member/decrypt.py <name> , interacts with Tap card, authenticates CLA, decrypts the file key, verifies the tag and renders the text to the OLED (or the non-text to disk).

7.3 Member Security Posture and Configuration

The Pi is saved. The I 2 C/SPI interface is enabled. The advertisement is based on services used. The crucial file access is restricted to Root only. The liveliness nonces are

signed in ATECC608A. The BME280 proveance can also be stored in conjunction with values in RTP Header. The AAD can also be used to limit values in BME280.

In the case study, it has been presented that the decryption constrained by the device takes place through the use of the class scoped key, and the RFID presence gate remedied the four common challenges faced by the teams cloud sharing the plaintext since classes have been blocked earlier, and the integrity breach has no capability to offer information.

8 Results

- **Encrypt and upload:** On the Pi, the file encrypted with aes-gcm security to be transmitted to Cloud comprising both cipher text and a small header was issued by encrypt. py.
- **Cloud is blind:** The blind output of the curl of a download (ciphertext value and the JSON header value): class, nonce, tag, wrap and checksums are exposed. The server has no ciphertext and no keys.
- **Tap to open:** Awkwardly Running decrypt.py displayed Ready -Tap card on the OLED. The Pi required tapping an enrolled RFID, which verified class, unwrapped the key, integrity was verified, and decrypted.
- **Text view:** Trident visuals If the file was text, it was viewed on the OLED pager, dashed to dec(name)
- **Security checks:** Unrecognised RFID element or not authorised class, no unwrap of the Access Denied. Modified header/ciphertext - Integrity Check Failed (no plaintext).
- **Rotating the key:** It was easy to use dual wraps to switch to a new aggregate key; there was no need to re-encrypt stored data.

Encryption/Upload Process, as shown in Figure 3 below: The Raspberry Pi encrypts and uploads file to cloud using AES-GCM encryption algorithm, generates a reduced size KAC header with the usage of classes nonce tag wrap hashes and transmits the same to the cloud, and the cloud responds back with the 200 response but doesn't read or view the plain texts or keys.

Figure 4 below shows that for the RFID-Gated Decryption Path, Once the authorized tap entry is established, the file(key) decryption request in the Pi is made upon verification with the class authorization using the aggregate key within the KAC and GCM tag, and the plaintext is written, and for the cases involving the non-authorized cards, non-authorized or Data Modification, the Access Denied/Integrity Failure message is received without revealing the plaintexts.

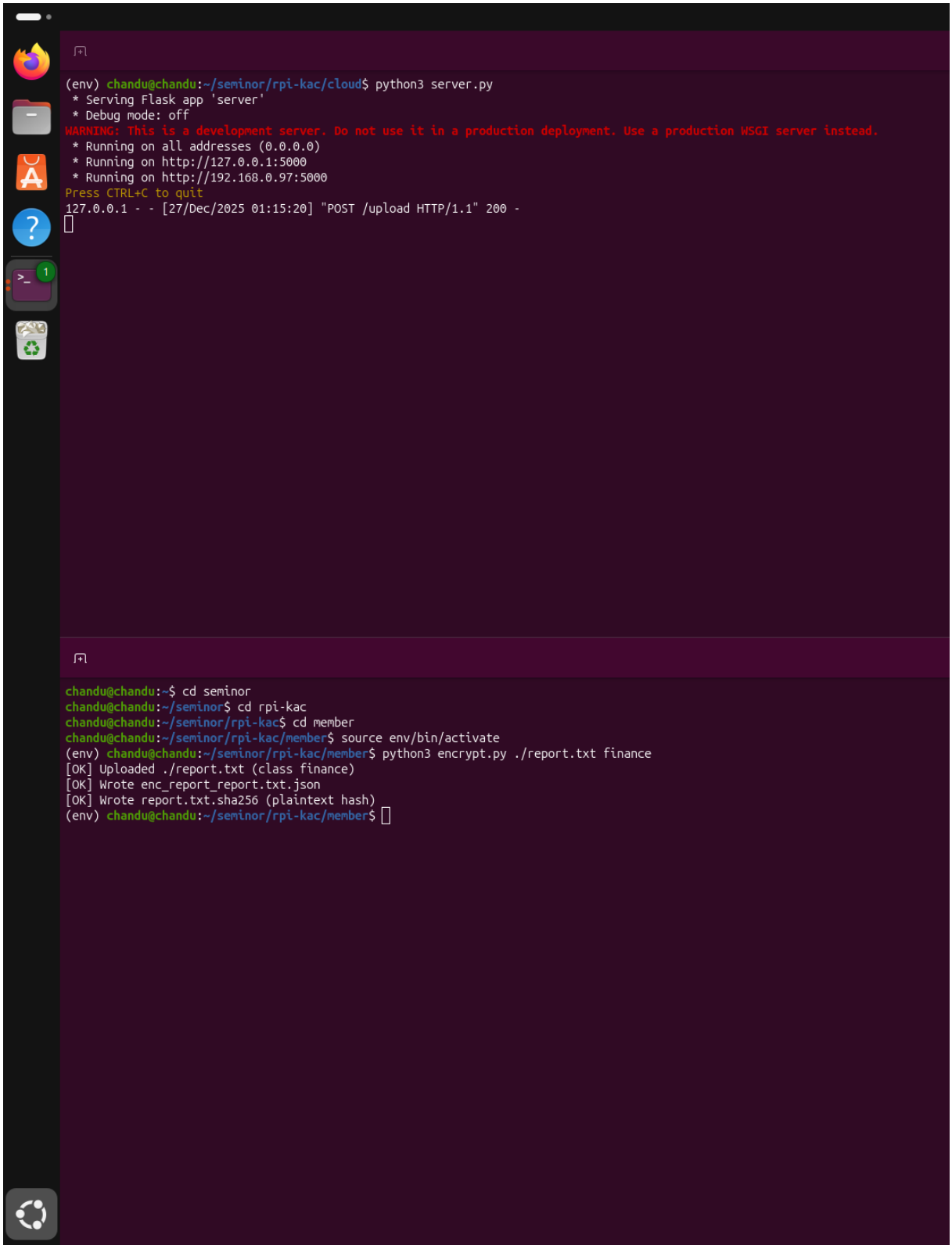

Figure 3: Member (Encrypt) — local AES-GCM + KAC wrap and upload.

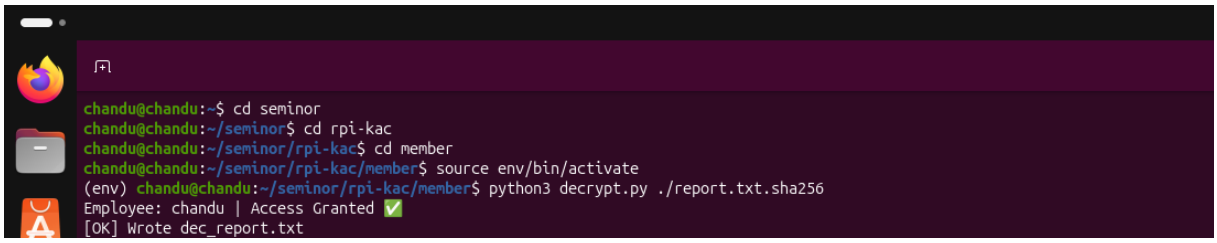

Figure 4: Member (Decrypt) — RFID-gated KAC unwrap and AES-GCM decrypt.

9 Research Question

1.How is it possible to turn on server-verified device attestation without negatively affecting usability?
This is important since the primary benefit of making a bind before downloading content is that it decreases chances of key theft, however it should not create extra workload to operators..

2.Which is the least amount of header that can be used defensively in terms of interoperability, audit, and integrity?
The importance of this is because metadata may leak class and equality information and management of fields by using TLS and AAD should not compromise verifiability.

3.What about aggregate key rotation and revocation in fleets that have zero downtimes?
This has significance to determine the safe cutovers (dual wraps, key IDs, rollback) unless re-encryption of saved information and stranding gadgets..

4.What presence characteristics do RFID optimally integrate into Ob the Pi (PIN, FIDO2 touch, time windows)?
This is important in that the strength of presence and the burden on the operator have to be decoupled such that there are unattended decrypts without damaging usability..

5.How do post quantum KEMs/signatures maintain the roles and flows of existing systems?
This is noteworthy because the hybrid headers will increase; also there might be an increase in the cost of computing; to which we require the migration routes that do not impact policy across devices...

6.What are the guidelines to the packaging, signature, and distribution of policy-as-code to edges?
This is required to have sound taxonomies of classes, expirations, constraints, verifiable updates and audit trail..

7.What are the most effective mitigations on to defend the I 2 C / SPI buses

and secure-element wiring on the Pi?
This is important as devices can be compromised over time in terms of trust of physical/bus probing; shielding, rate-limited, and tamper-evidence should be reviewed..

8.To what extent media is useful environmental provenance (e.g. BME280 data), and how is it to be secured?
The same applies to achieve a balance between forensic value and privacy, so one has to decide whether to bind context in AAD or restrict it to local logs...

10 Conclusions

One of the projects involves a practical hardware-backed design of secure cloud sharing where plaintext is never transmitted over the network, and only ciphertext and a small header are sent to the cloud. The authorization is Key-Aggregate Cryptosystem, which means that for a given single fixed-sized aggregate key, only classes of choice can be accessed and this is extremely fast to rotate and revoke and does not involve re-encryption of data. All files are end-to-end AES-GCM encrypted and nonce Hedged with a new key, and an associated data policy control bound down on header Tamper. The presence of an enrolled RFID tap triggers decryption and an SSD1306 OLED shows and audits responses. Attestation and BME280 context enhance optional ATECC608A to produce more lively and attestable devices that do not disclose their secrets or increase cloud trust.

Evidence-wise, there was proper class and proper RFID decryption every time, or invalid RFID or improper class was denied before any unwrap and a modification in anything in the header or ciphertext would make the tag fail without being able to recover the plaintext. Functionally, key rotation was only required to rotate one more aggregate key, and dual wrap cutovers made it possible to make zero-downtime transitions. The cloud was also small and white and diminished the probability of breaches and made it easy to comply. Overall, it is possible to reproduce the system because it is lightweight and works well in small groups or IoT networks who need device bound decryption and a class-scope policy. Future work includes default server-verified attestation, more shrinking in the size of the headers in mandatory TLS scenarios, mass-scale management of key lifecycle, and hybrid post-quantum migrations without role or workflow changes.

References

[1] C.-K. Chu, S. S. M. Chow, W.-G. Tzeng, J. Zhou, and R. H. Deng, "Key-aggregate cryptosystem for scalable data sharing in cloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 2, pp. 468–477, 2014, Available: https://doi.org/10.1109/TPDS.2013.112. DOI: 10.1109/TPDS.2013.112.

[2] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of the IEEE Symposium on Security and Privacy*, Available: https://doi.org/10.1109/SP.2007.11, 2007, pp. 321–334. DOI: 10.1109/SP.2007.11.

[3] M. Ambrosin, M. Conti, R. Lazzeretti, M. M. Rabbani, and S. Ranise, "Collective remote attestation at the internet of things scale: State-of-the-art and future challenges," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2447–2461, 2020, Available: https://doi.org/10.1109/COMST.2020.3008879. DOI: 10.1109/COMST.2020.3008879.

[4] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying iot security: A comprehensive survey of iot vulnerabilities and defenses," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2702–2733, 2019, Available: https://doi.org/10.1109/COMST.2019.2910750. DOI: 10.1109/COMST.2019.2910750.

[5] A. Juels, "Rfid security and privacy: A research survey," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 381–394, 2006, Available: https://doi.org/10.1109/JSAC.2005.861395. DOI: 10.1109/JSAC.2005.861395.

[6] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016, Available: https://doi.org/10.1109/JIOT.2016.2579198. DOI: 10.1109/JIOT.2016.2579198.

[7] S. Yu, C. Wang, K. Ren, and W. Lou, "Secure, scalable, and fine-grained data access control in cloud computing," in *Proceedings of IEEE INFOCOM*, Available: https://doi.org/10.1109/INFCOM.2010.5462175, 2010, pp. 1–9. DOI: 10.1109/INFCOM.2010.5462175.

[8] M. N. Khan, A. Rao, and S. Camtepe, "Lightweight cryptographic protocols on iot-constrained devices: A survey," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4132–4156, 2021, Available: https://doi.org/10.1109/JIOT.2020.3026493. DOI: 10.1109/JIOT.2020.3026493.

[9] E. Rescorla et al., *The datagram transport layer security (dtls) protocol version 1.3*, RFC 9147, Available: https://doi.org/10.17487/RFC9147, 2022. DOI: 10.17487/RFC9147.

[10] M. Dworkin, "Recommendation for block cipher modes of operation: Galois/counter mode (gcm) and gmac," National Institute of Standards and Technology (NIST), Tech. Rep. SP 800-38D, 2007, Available: https://doi.org/10.6028/NIST.SP.800-38D. DOI: 10.6028/NIST.SP.800-38D.