AKESO: Bringing Post-Compromise Security to Cloud Storage

William & Mary — Chartered 1693

Lily Gloudemans algloudemans@wm.edu
Pankaj Niroula pniroula@wm.edu
Aashutosh Poudel apoudel01@wm.edu
Stephen Herwig smherwig@wm.edu

/etc/lab — Extending Trust in Computing Lab

Confidentiality Risks in Cloud Storage

- Data disclosure to law enforcement
- Data leaks due to internal and external attacks
- Key rotation does not re-encrypt existing objects
- Questionable key rotation and revocation strategies

Data Encryption in Cloud Storage

Client-side Encryption	Server-side Encryption
✓ Encryption keys stay with client	✓ Cheap and scalable storage
⚠ Key management complexities	⚠ Key rotation without object re-encryption
⚠ High ingress and egress during key rotation	⚠ Susceptible to key leakage

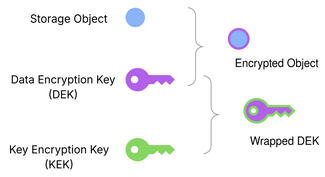


Fig 1: Object encryption in Google Cloud Storage

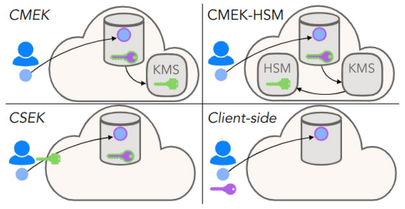


Fig 2: Comparison of Google Cloud Storage Encryption Options
CSEK: Customer-Supplied Encryption Key
CMEK: Customer-Managed Encryption Key
HSM: Hardware Security Module

Cloud Storage Encryption: Practices vs Implications vs AKESO

DEKs exist in the cloud during encryption

At risk of adversary access in a vulnerable cloud

Group key agreement with Asynchronous Ratcheting Tree (ART)

- ✓ DEKs never exposed to the cloud
- ✓ Post-compromise security
- ✓ Alleviates key management complexities

Key rotation updates only the KEK without re-encryption

Leaked DEKs can decrypt unmodified objects

Updatable encryption with nested AES

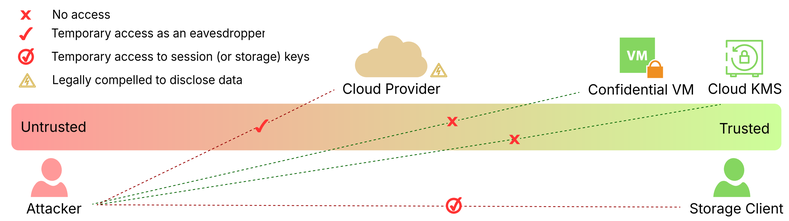
- ✓ Untrusted cloud functions can re-encrypt objects
- ✓ Efficient re-encryption with small update tokens
- ✓ Reduces data ingress and egress on key rotation

Comparable performance to existing cloud storage encryption

No changes required at cloud

- ✓ Utilizes cloud services like Pub/Sub, Cloud functions
- ✓ Implemented in client-side software (`gcsfuse`)

Threat Model



Design

Design Goals:

✓ Post-Compromise Security ✓ Efficient re-encryption ✓ Compatibility and Transparency

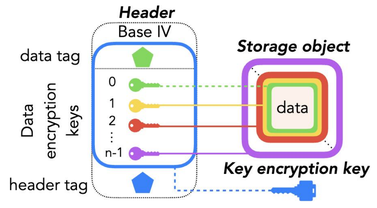


Fig 3: Updatable encryption using nested AES

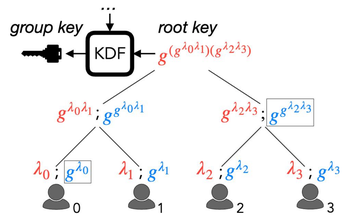


Fig 4: ART with four group members. Red Keys are private and Blue Keys are public

Nested AES
To achieve an efficient symmetric encryption scheme capable of key rotation by an untrusted cloud function, we use nested AES-256 similar to the scheme presented in [1]

Why ART?
- Post-Compromise Security
- Group key rotation time scales logarithmically with group members
- Any member can rotate the group key; all others can securely calculate

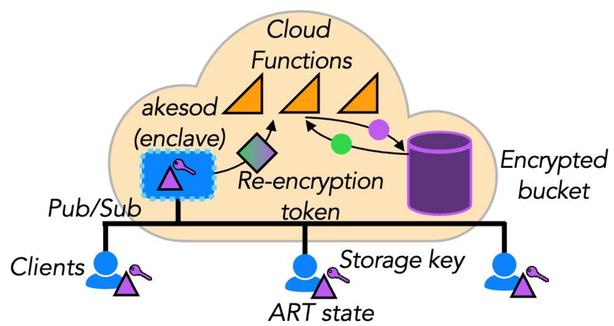


Fig 5: High-level Architecture of AKESO

Components Used:

- **Buckets** — Cloud storage endpoint to host the files
- ***akesod* - Group Orchestrator** — Trusted daemon that initiates group setup and re-encryptions
- **gcsfuse** — Cloud storage client
- **Cloud Function** — Serverless function triggered by **akesod** to re-encrypt all objects in a bucket
- **Pub/Sub** — Channel for broadcasting the ART setup and key update messages
- **Confidential VM** — AMD SEV VM that hosts **akesod** in the cloud

Evaluations

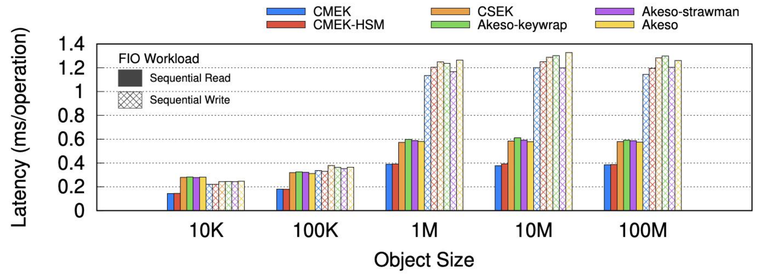


Fig 6: Latency of read and write operations for encrypted cloud storage using different strategies*

* ***Akeso-strawman***: **akesod** re-encrypts the objects itself, rather than using cloud functions.
* ***Akeso-keywrap***: AKESO analog to CSEK, in which **akesod** rewraps the encryption key but does not re-encrypt the object.

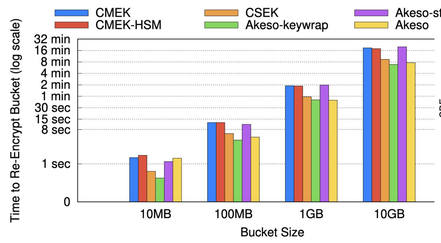


Fig 7: Time to re-encrypt a bucket of varying sizes, where each bucket object is 2M

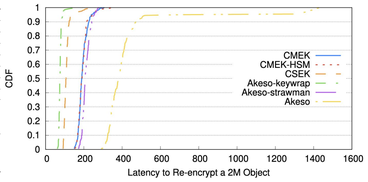


Fig 8: Latency (including network overhead) to re-encrypt a 2M object, varying the re-encryption method

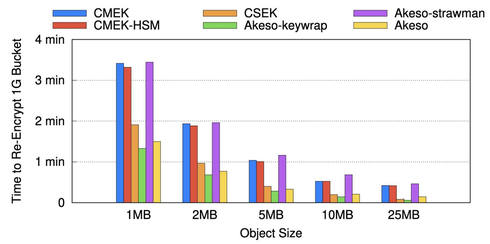


Fig 9: Time to re-encrypt a 1G bucket, varying the size of the objects in the bucket

Takeaways
- Filesystem operation performance of AKESO is comparable to existing encryption options
- Storing data as fewer, larger objects allows for more efficient re-encryption of a bucket
- Re-encryption performance of AKESO is comparable to existing encryption options
- As bucket size increases, AKESO consistently re-encrypts faster than the strawman approach

Environment	Akeso	Strawman
Confidential VM	47.231	118.239
Non Confidential VM	44.907	109.366

Table 1: Time (in seconds) to re-encrypt a bucket of size 1G on confidential vs non-confidential VM

Takeaway
The security guarantees of a confidential VM incur low overheads

Takeaways
- The most significant cost is that of running **akesod** in a confidential VM
- Running **akesod** on-premises incurs egress costs that would scale notably with bucket size

AKESO	Cloud Enclave	On-Premises
Compute	144.71	0
Storage	3.9	3.9
Pub/Sub	0	0.12
Data Egress	0	12
Cloud Function	0.51	0.51
Total	149.12	16.53

Table 2: Monthly cost(USD) breakdown for running **akesod** on a Cloud Enclave vs. On-premise Server

References

[1] D. Boneh, S. Eskandarian, S. Kim, and M. Shih, "Improving speed and security in updatable encryption schemes," in International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), 2020.
[2] K. Cohn-Gordon, C. Cremers, L. Garratt, J. Millican, and K. Milner, "On ends-to-ends encryption: Asynchronous group messaging with strong security guarantees," in ACM Conference on Computer and Communications Security (CCS), 2018.