Geocryptark: a secure digital ark linked to a geographic coordinate and protected by cryptography.

Patrick
patrick-mns@hotmail.com

June 2025

Abstract

Introducing a cryptographic architecture that allows only users who know the correct location (latitude and longitude) and a common password to decrypt protected content. This approach is ideal for applications that rely on geographic access restrictions, such as treasure hunt games, augmented reality experiences, and interactive map-based narratives.

1 Introduction

Traditional location-based access control systems generally rely on geographic coordinates, tokens, or other centralized authentication mechanisms — which implies trusting third parties for validation. This approach presents significant limitations, especially when the goal is to guarantee autonomy, privacy, and resistance to censorship or external manipulation.

I propose a decentralized alternative where access to content is linked to a specific geographic point, and validation can occur directly between peers (*peer-to-peer*), without the need for trusted servers. To do this, I combine geographic coordinates with a password to derive cryptographic keys capable of unlocking protected data only at authorized locations.

This is not a formal academic paper but a practical proposal I called **Geocryptark**. The name consists of three central elements: *Geo* for geography; *crypt* for cryptography; and *ark*, evoking the idea of a secure chest or vault where data is protected.

2 Formalization

Given a secret object O to be stored securely, we use a cryptographic approach based on authorized locations L_i, combined with a common password P and an additional entropy value *salt*. The complete procedure is described below:

Inputs

- O: data to store in the vault.
- $L_i = (lat, lng)$: one or more points on the map.
- P: common password (analogous to the vault key).
- *salt*: value to increase entropy.

Hash Generation

For each location L_i, concatenate its coordinates with the *salt* and password P, forming the message m:

$$m = L_i \parallel salt \parallel P$$

We apply the hash function:

$$h = \text{SHA256}(m)$$

Key Derivation

The symmetric key K_{L_i} is derived by:

$$K_{L_i} = \text{PBKDF2}(h, salt_{\text{const}}, n, \text{SHA256})$$

Data Encryption

Before applying location-based and distributed encryption, we protect the original message O with an initial symmetric encryption layer using only the common password P and a *salt*:

$$O = \text{AES\_Encrypt}\left(O, \text{KDF}(P \parallel \textit{salt})\right)$$

We generate a random session key:

$$K_{\text{sess}} \xleftarrow{R} \{0,1\}^{256}$$

And encrypt the message O again:

$$C = \text{AES-GCM\_Encrypt}(O, K_{\text{sess}}, IV)$$

Session Key Wrapping

For each authorized location L_i, we encrypt the session key with its respective derived key:

$$W_i = \text{AES-GCM\_Encrypt}(K_{\text{sess}}, K_{L_i}, IV_i)$$

Decryption

A client who has:

- The correct coordinate L_i,
- The common password P,

can generate the key K_{L_i}, decrypt W_i to obtain K_{sess}, and finally recover the message O from C.

3 Artifact Distribution

- The *salt* used;
- The encrypted data C and its initialization vector (IV);
- The list of wrapped keys W_i and their respective IVs.

Distribution Methods

There are two main approaches to distribute the encrypted artifacts:

1. Intermediate server model: `peer → server ← peer`

In this approach, the server acts as a repository of the encrypted data generated by a peer. It stores the artifact and compares hashes received from other peers to validate accesses. The key point here is that the server never has access to the latitude, longitude, or secret password — only to the hash derived from these data. Even if the hash is intercepted, it does not allow access to the content since the message still depends on the session key and password. This strategy removes the need to trust the server, maintaining system security. In a simple analogy: the server acknowledges that you know where the vault is and hands over the vault, but to open it, you still need the key.

2. Peer-to-peer model: In this approach, the peer can generate all the encrypted data locally and distribute it directly through any channel (P2P networks, QR Codes, files, etc.), without needing a server.

In this scenario, the main challenge becomes protection against brute-force attacks. Since there is no server intermediating, there is no direct control over unlocking attempts. In hybrid contexts, the server can act only to apply rate limiting or temporary lockout mechanisms, helping protect the artifact.

This architecture follows the principle that the "vault" can be accessed by multiple peers simultaneously. Once the secret is discovered, the content can be "looted" — which, depending on the application goal (e.g., games or challenges), may be desirable.

4 Conclusion

The proposed system combines high-security symmetric cryptography with geographic restrictions, creating an access control mechanism that depends both on knowledge (the password) and presence (the location). This approach eliminates the need to trust third parties, enabling fully peer-to-peer implementations where data remains protected even against malicious or compromised servers.

The proposal, named **Geocryptark**, offers a versatile solution for contexts where access should be conditioned on presence at a specific coordinate — whether in playful applications such as games, distribution of secret content in physical locations, or authentication systems in controlled environments.

Although this work is not a formal academic paper, it serves as a basis for practical experimentation and future improvements. The central idea is to demonstrate that space, secrecy, and cryptography can be linked simply yet effectively.

References

[1] Niels Ferguson, Bruce Schneier, Tadayoshi Kohno. *Cryptography Engineering: Design Principles and Practical Applications*. Wiley Publishing, 2010.

[2] Jonathan Katz, Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC, 2nd edition, 2014.

[3] National Institute of Standards and Technology (NIST). *Recommendation for Password-Based Key Derivation: Part 1 – PBKDF2*. NIST Special Publication 800-132, 2010. `https://doi.org/10.6028/NIST.SP.800-132`

[4] Morris Dworkin. *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*. NIST Special Publication 800-38D, 2007. `https://doi.org/10.6028/NIST.SP.800-38D`

[5] Internet Engineering Task Force (IETF). *JSON Web Encryption (JWE)* — RFC 7516. `https://tools.ietf.org/html/rfc7516`

[6] ISO/IEC 18033-3:2010. *Information technology — Security techniques — Encryption algorithms — Part 3: Block ciphers*. (Defines the concept of key wrapping via symmetric cryptography.)