Json Fine Grained Access

(proposed specification)

Jinling Zhang,Cheng Jiang and Lingling Ji

IETF 119, Brisbane

March 18, 2024

Why It Matters

- Traditional access control methods, such as Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC), although protecting data security to some extent, gradually show their limitations when dealing with complex and dynamic data access requirements.
- To address this issue, this paper proposes a JSON-based fine-grained access control method that can be applied to various scenarios such as web services, cloud computing, and the Internet of Things.

Proposed Solution

JSON-FA (JSON-based Fine-Grained Access Control) data format is a standardized format used to initiate access requests to access control systems.mainly consists of a JSON object that contains the requested access resources and their fine-grained access conditions:

- requestId: The identifier of the request
- subject: Identifiers representing the access subject.
- operation: indicates the action identifier of the request body, such as "read", "write", "update", and so on.
- resource: Indicates the identifier of the accessed resource.
- condition: Refers to the fine-grained attributes of the access subject

Proposed Solution

JSON-FA process flow：

- The client sends authentication information to the server and receives a JWT token.
- The request sent by the client includes a JWT token, and the payload of the token contains information about the accessing subject, accessed resources, and attributes
- Resource servers set fine-grained access control policies for different resources, which are represented by an Access Control Tree (Access Tree).
- The server verifies the validity of the token, parses the payload information, and extracts the access information and attribute information.

Proposed Solution

JSON-FA process flow

- Based on the extracted information and the predefined access control policy, it is determined whether there is a corresponding policy. If there is, the access conditions are checked to see if they are met. If the conditions are met, access is granted; otherwise, access is denied.
- The resource server encrypts the requested resource using the CP-ABE algorithm. The encryption process incorporates the access control policy corresponding to the accessed resource. The encrypted ciphertext is then sent back to the client.
- The client generates a decryption private key based on the set of attributes included in the request using the CP-ABE algorithm. After receiving the encrypted ciphertext, the client verifies that the attributes in the attribute set can satisfy the access control policy. Only if the attributes meet the policy can the client decrypt and access the resource.