# Chained Credentials-Based Authorization (2CBA)

Igor Zboran
izboran@gmail.com

***Abstract*—User-Managed Access (UMA) [1, 2] is an authorization framework built on the top of OAuth 2.0 protocol that allows users to delegate access to their resources to other users. UMA manages this trick by using a sophisticated delegation model. The problem is that this model uses authorization flows when one resource server needs to access another resource server in order to fulfill the client request.**

**The authorization trail is the intrinsically recorded transaction flow of the authorization process. It comprises cryptographically chained blocks of data bearing chronological tamper-resistant records of all their possessors and the changes that have been made by them. Given that, the authorization trail can carry both coarse-grained and fine-grained authorization data. Notably, this is useful in the healthcare industry and financial sector.**

**A nested, chained MAC construction (e.g., HMAC) is used to ensure the authenticity and integrity of the credentials. All sensitive information is encrypted to preserve confidentiality. The authorization decision relies on a real-time examination of the sequence of credentials by the authorization server. The 2CBA concept is compatible with existing OAuth 2.0 based protocols.**

## I. Introduction

Bearer tokens are easy to use and easy to integrate with any client, server, and mobile device. Once they have been minted on the authorization server, they don't require additional processing on the client, and token validation on the resource server is trivial. While existing technology standards are acceptable for many use cases, they have been harder adopted in the healthcare industry and financial sector, where a higher degree of authenticity and integrity of the authorization flow is required. To overcome these restraints, additional data alongside the access token should be provided. The authorization trail mechanism aims to provide these data by recording detailed information during the authorization flow. The resulting authorization trail increases the level of security, while ensures compliance with financial and legal regulations by documenting all actions and events of the authorization flow.

## II. Concept

The 2CBA concept is based on verifiable tamper-resistant sequence of credentials created by authenticated principals (authorization server, client, resource server) during the respective stages of the authorization process. The sequence of credentials carries information in the form of a sequence of records organized into blocks. Each credential comes from an individual principal—a credential issuer. Claims and credentials are chained using the MAC value of the previous record.

### A. List of Claims

To create a list of claims, we use the HMAC chaining construct $MAC$ = HMAC($K$, HMAC($MAC$, $m$)), broken down into individual MACs,

$MAC$ = HMAC($MAC$, $m$)
$MAC$ = HMAC($K$, $MAC$)

which forms the basis of the credential chaining mechanism.

To simplify notation, we use the Double HMAC construct—a nested HMAC function, denoted by DHMAC, that takes three inputs ($K$, $MAC$, $m$) and outputs a message authentication code,

$MAC$ = DHMAC($K$, $MAC$, $m$) = HMAC($K$, HMAC($MAC$, $m$))

where $K$ is the secret key, $MAC$ is the input message authentication code, and $m$ is the message to be authenticated.

### B. Sequence of Credentials

The credentials are issued by principals. Each principal must be registered at the authorization server.

Each credential contains four mandatory claims:

- The random NONCE to prevent replay attack.
- The timestamp of when the credential was created.
- The URI that identifies who created the credential.
- The MAC value of the last claim copied from the previous credential.

Aditional claims can be added by the respective principal at any time until the credential is sent to the next principal.

The sequence of credentials is terminated by the terminal, final MAC value of the sequence.

### C. Verification

The authorization trail is reviewed in real-time by the authorization server that acts as an examinator. Given that, the resource server uses the token introspection endpoint of the authorization server to verify the authorization trail and look up relevant information about the authorization process. The verification system of the authorization server is able to recreate MACs of the individual blocks of the authorization trail using the shared secrets of the authorization flow participants. Thus, if any block of the authorization trail was altered, then the authorization trail tampering can be detected by comparing the recreated MACs with the authorization trail MACs.

## III. Use Cases and Concrete Mechanisms

A User-Managed Access authorization flow: client->RS->AS=>RS=>client->AS=>client->RS->AS=>RS=>client

-> request
=> response

### A. Permission Tickets

TBD

### B. Managing Resources

TBD

### C. Managing Resource Permissions

TBD

## IV. Conclusion

By utilizing simple cryptographic techniques, we can build the authorization trail that, mitigates the risk that an adversary could gain illegal access to private information by stealing a token. As part of the broader applicability of the authorization trail, the recorded data can later be used by government agencies and businesses for forensic analysis and compliance verification.