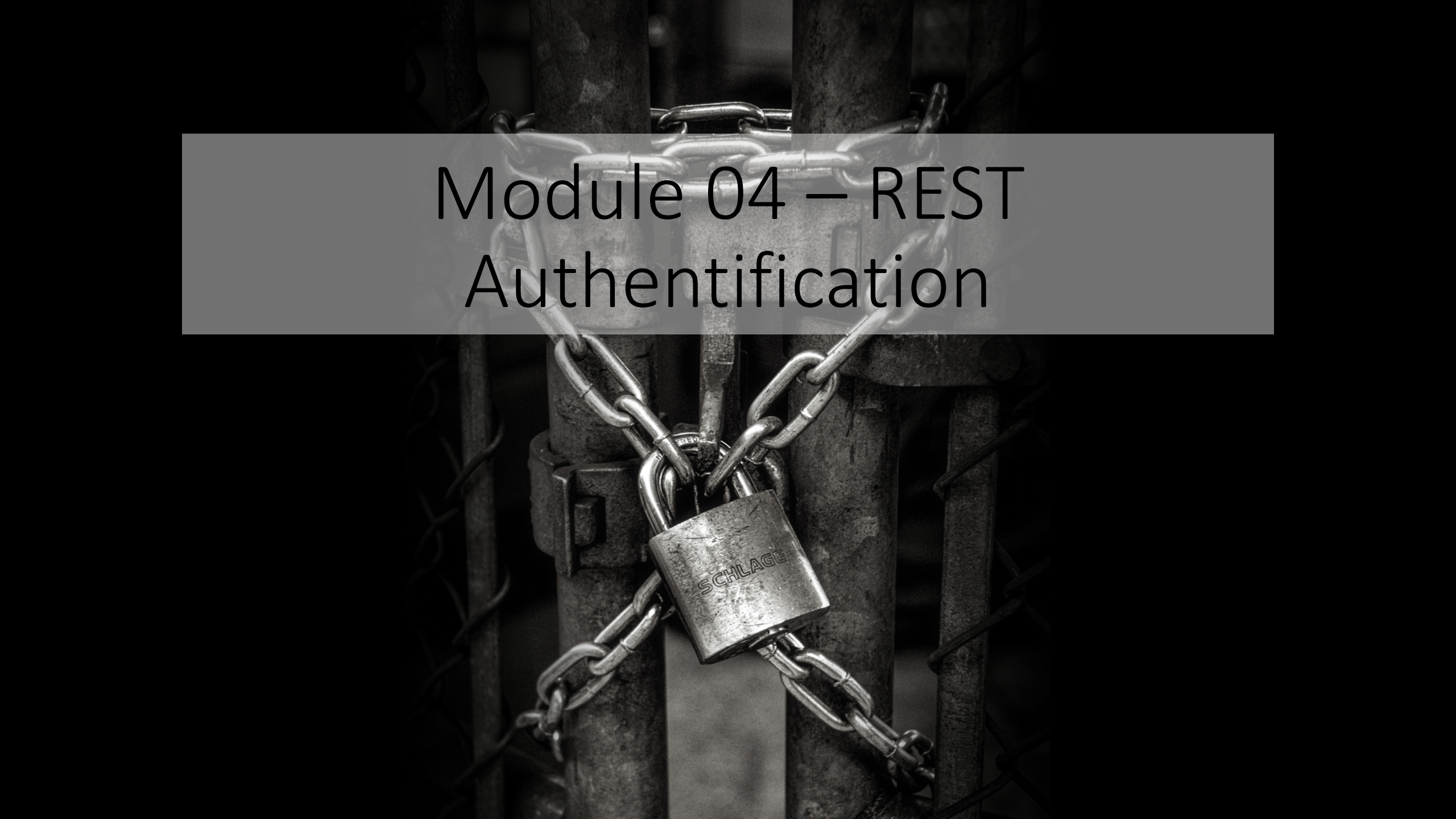
Module 04 – REST Authentification



Objectif

- Identification / Authentification / Autorisation
- Quelques moyens d’authentification
- Implantation en C#

Identification / Authentification / Autorisations

- L’identification est une phase établissant l’identité d’un utilisateur ou d’un système :
  - Nom d’utilisateur
  - Adresse courriel
- L’authentification est une phase subséquente validant l’identité :
  - Mot de passe
  - Certificat
  - Code envoyé par courriel, SMS, une application ou un jeton

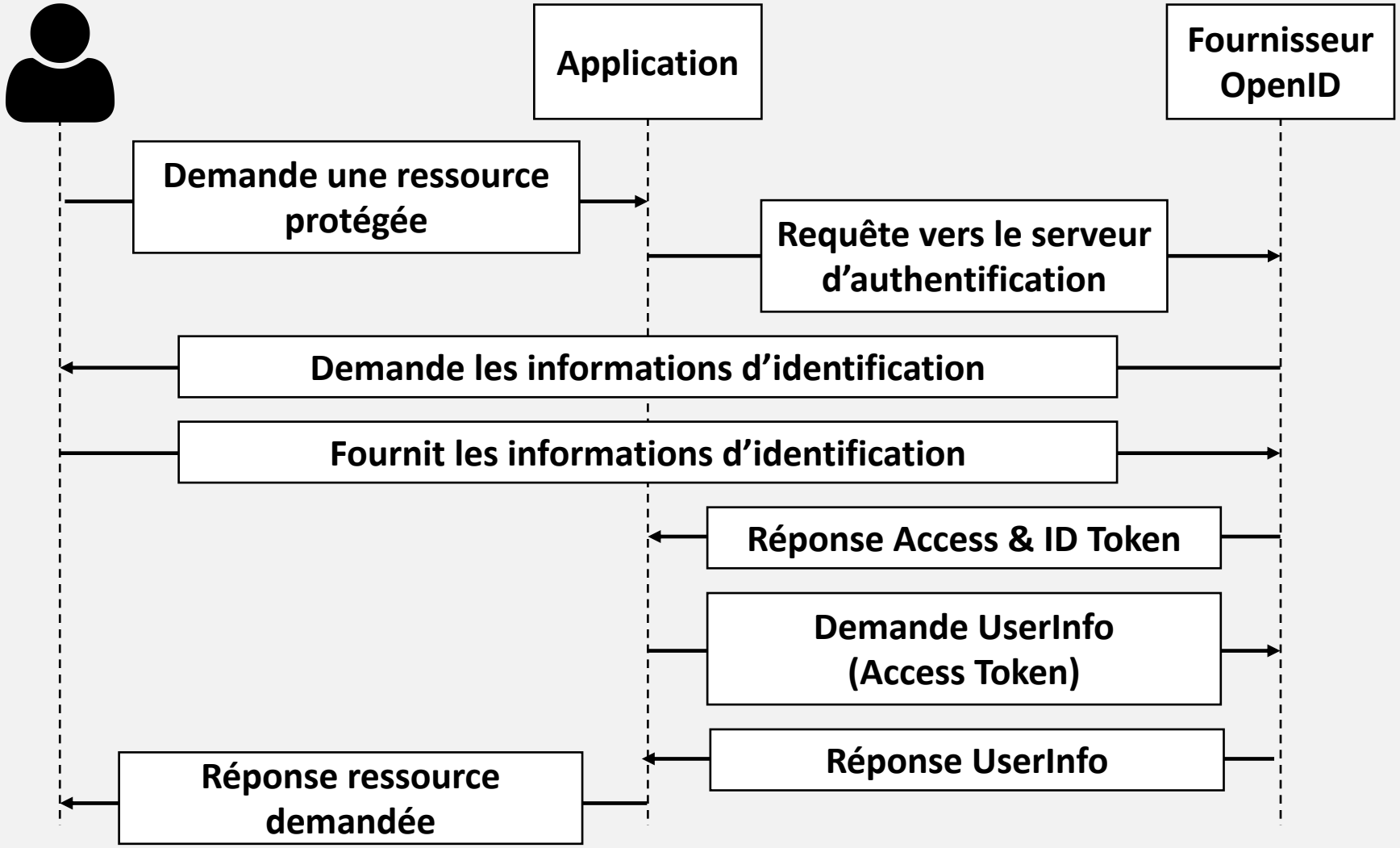
Identification / Authentification / Autorisations

- Pour renforcer la sécurité, l'authentification peut être effectuée à partir de plusieurs moyens :
  - Mot de passe + code SMS
  - Mot de passe + jeton d'authentification RSA
- Les autorisations concernent les actions que l'utilisateur ou le système peut effectuer dans l'application
  - Si basées sur les rôles, on parle de RBAC (Rôle Base Access Control)

Quelques moyens d’authentification

- Nom d’utilisateur et mot de passe : le plus simple et le plus classique
- NTLM / Kerberos : souvent utilisé dans les intranet en entreprise
- OAuth2 : délégation d’authentification
- OpenID connect (OIDC)
  - Basé sur OAuth2
  - Identification d’un utilisateur -> ID Token au format JWT
  - Peut service pour faire du Single Sign-on (SSO)

OpenID Connect – Flux première authentificaiton



https://openid.net/specs/openid-connect-basic-1_0.html

Exemple API avec clef – C#

- En C#, on peut créer une méthode qui sera appelée à chaque requête en utilisant ce que le cadriciel appelle un filtre
- Il existe plusieurs mécanisme pour le mettre en place, ici on va utiliser un attribut personnalisé que nous allons ajouter au contrôleur à protéger

Exemple API avec clef – C#

```
[AttributeUsage(AttributeTargets.Class | AttributeTargets.Method)]
public class ApiKeyAttribute : Attribute, IAsyncActionFilter {
    const string clefValide = "59604896-66a4-4a9b-8f7b-94a5d16bbdaf";

    public async Task OnActionExecutionAsync(ActionExecutingContext p_context, ActionExecutionDelegate p_next) {
        StringValues clefAPI;

        if (!p_context.HttpContext.Request.Headers.TryGetValue("clefAPI", out clefAPI)) {
            p_context.Result = new UnauthorizedResult();
            return;
        }

        if (!clefValide.Equals(clefAPI)) {
            p_context.Result = new UnauthorizedResult();
            return;
        }

        // Excécute la suite des filtres
        await p_next();
    }
}
```

Exemple API avec clef – C#

```
[ApiKey()]
[Route("api/[controller]")]
[ApiController]
public class ValuesController : ControllerBase
{
    // GET: api/<ValuesController>
    [HttpGet]
    public IEnumerable<string> Get()
    {
        return new string[] { "value1", "value2" };
    }

    // [...]
}
```

Références

- https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity-api-authorization?view=aspnetcore-5.0
- https://openid.net