ECE PARIS
ÉCOLE D'INGÉNIEURS

Pare-feu

2019/2020

APERCU

Dans ce TP vous serez amenés à mettre en place un pare-feu pour sécuriser l'accès à l'API.

OBJECTIFS

1. Mettre en place un middleware
2. Créer un pare-feu

1. Mise en place d'un middleware

Dans ce TP, nous allons mettre en place un pare-feu afin de sécuriser l'accès à l'API. Avant de procéder à la suite du TP, veuillez créez une table **users** dans laquelle seront les colonnes **username** avec le type **varchar** de 255 caractères et une colonne **apikey** avec le type **varchar** de 255 caractères.

Nous souhaitons donc vérifier à chaque requête si une clef API est présente dans l'**en-tête** et si cette clef est correcte. Mais avant de mettre en place cette vérification, il faut d'abord créer ce qui est communément appellé un "middleware". Avec la librairie Express, chaque élément permettant de traiter l'objet de la requête HTTP est un middleware; c'est à dire une fonction appelée lorsque la requête doit être traitée selon certaines conditions. Par exemple, chaque route créée jusqu'ici correspond à un middleware associé à deux conditions: l'URL et le type de la requête HTTP.

Nous allons donc tout d'abord créer un middleware qui sera traité qu'importe le type de la requête et son URL:

```
app.use(function(req, res, next) {

});
```

Attention: Si l'on souhaite que ce middleware soit traité avant toute route, il faudra le placer avant les routes dans le code.

2. Création du pare-feu

Maintenant que le middleware est créé, il faut mettre en place le pare-feu. Le pare-feu aura pour rôle de vérifier que l'option **X-Auth-Token** dans l'entête de la requête est bien présente et que l'option correspond bien à une entrée dans la base de données. Si les conditions ne sont pas remplies, alors le pare-feu renverra le message "Accès refusé".

```
app.use(function(req, res, next) {
    if ("x-auth-token" in req.headers) {
        let token = req.headers["x-auth-token"];
        let query = `SELECT * FROM users WHERE apikey='${token}'`;

        db.query(query, function(err, result, fields) {
            if (err) throw err;

            if (result.length > 0) {
                next();
            }
            else {
                res.send("Access denied");
            }
        });
    } else {
        res.send("Access denied");
    }
});
```

Remarque: La fonction **next** permet de passer au prochain middleware, donc dans notre cas, cela permet d'appeler la route associée à la requête.