



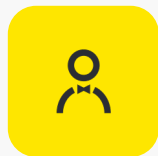
# Аутентификация и авторизация на платформе .NET

С использованием SSO  
на основе Keycloak



# О нас

Raiffeisen .NET Community



Развитые навыки разработки приложений использующих или расширяющих платформу .NET.



Должность, профильное образование и желание развивать экосистему .NET.



Использование технологий и приложений экосистемы .NET с открытым исходным кодом.

Это объединение  
сотрудников от  
разработчика до  
бизнес-эксперта

# Цели

## Обоз необходимых моделей авторизации

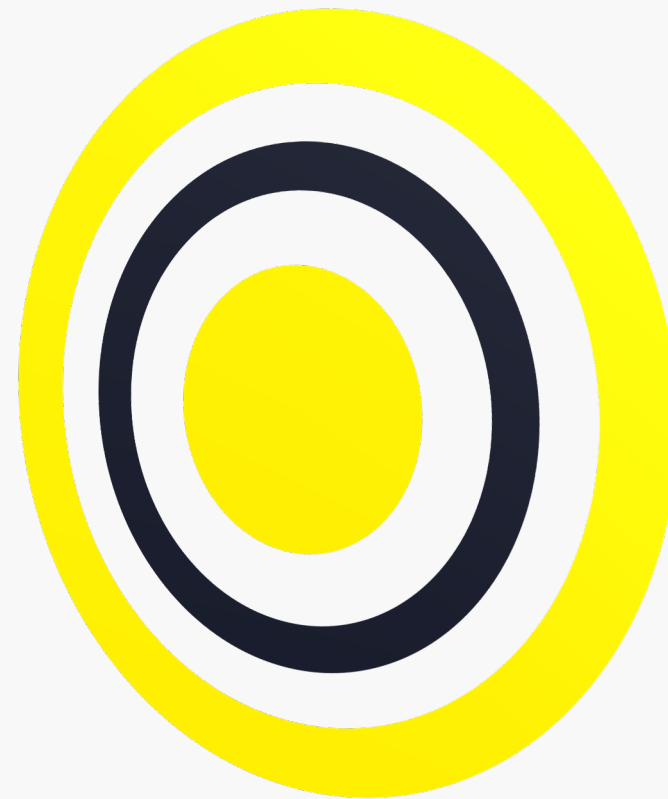
RBAC/ABAC/ACL когда какую применять и с чего не следует начинать проектирование авторизации вашего приложения.

---

## Осветить возможности платформы

Продемонстрировать возможности платформы по использованию различных моделей авторизации: RBAC/ABAC/ACL, в том числе при использовании IDP Keycloak

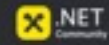
---



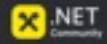
# Оглавление



**Модели  
управления  
доступом**



**Авторизация с  
использованием  
LDAP и Kerberos**



**Авторизация с  
использованием  
Keycloak**



**Итоги**



# Модели управления доступом



# Классификация

## Моделей управления доступом

- Дискретное управление доступом – **DAC, ACL**
- Мандатное управление доступом - **MAC**
- Управление доступом на основе ролей - **RBAC**
- Управление доступом на основе атрибутов - **ABAC**
- И другие: IBAC, CBAC, NBAC, ZBAC...

## В ASP.NET Core

- Простая авторизация
- На основе ролей - **RBAC**
- На основе утверждений - **ABAC**
- На основе ресурсов – **DAC, ACL**
- На основе политик – **ВСЁ сразу**

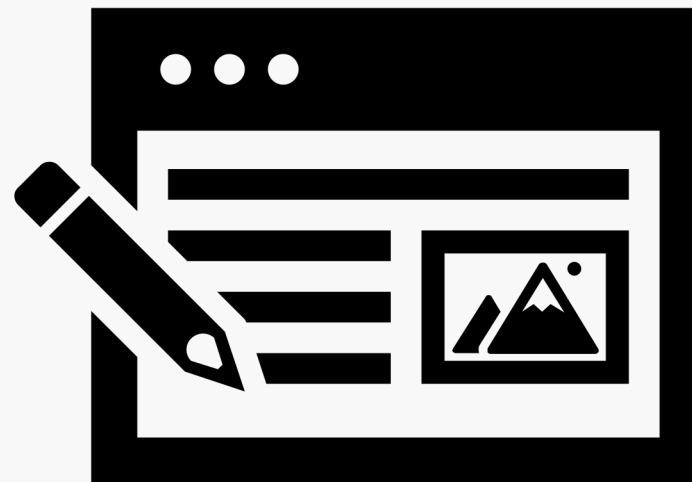
# На основе ролей

RBAC



Роли

Просмотр



# На основе ролей

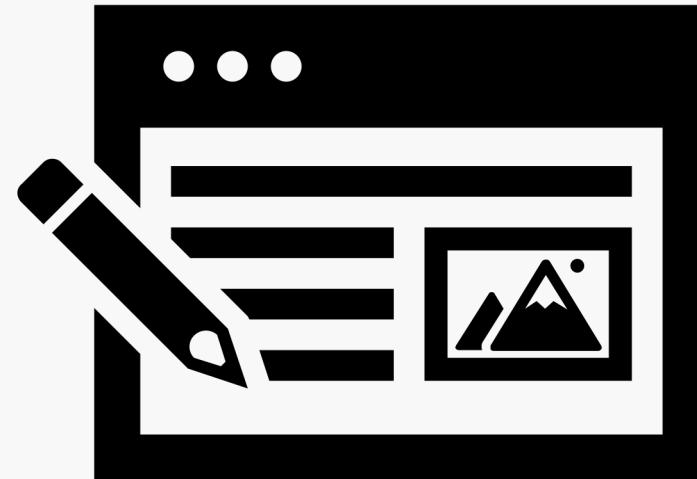
RBAC



Роли

Редактирование

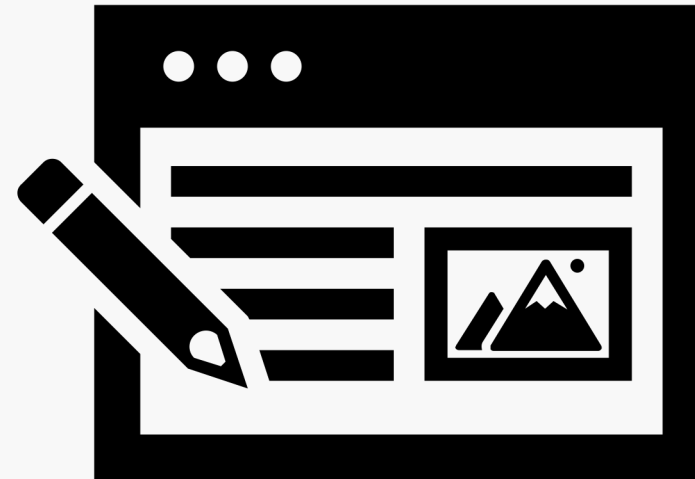
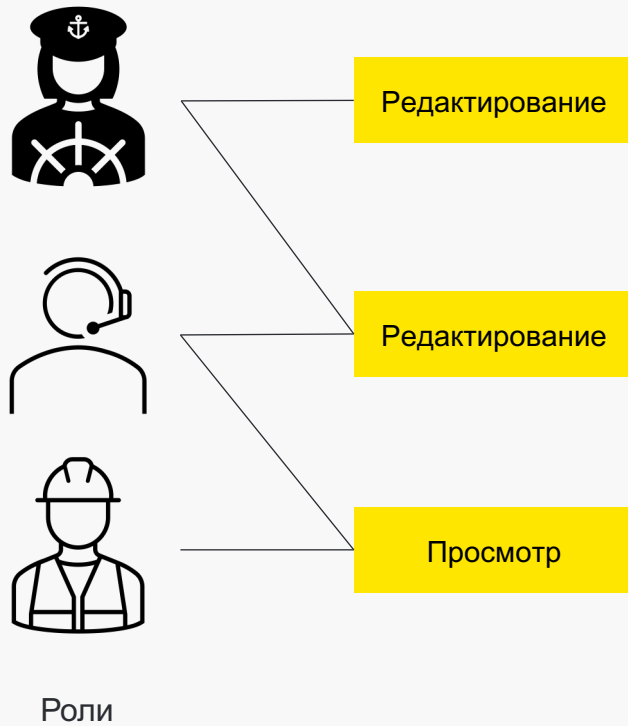
Просмотр





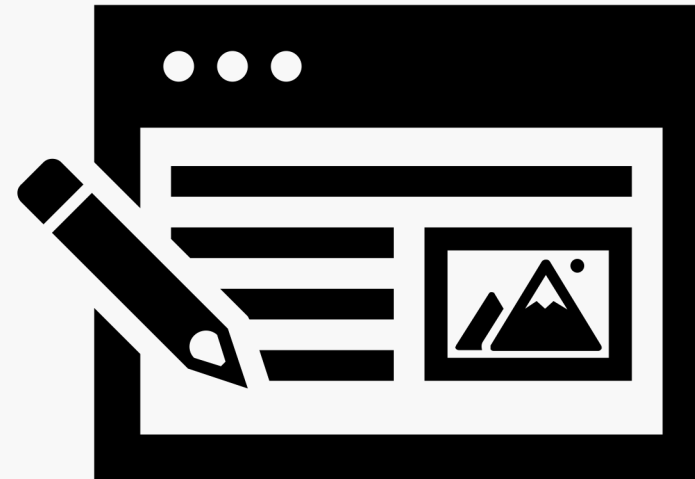
# На основе ролей

RBAC



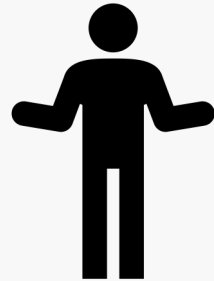
# На основе атрибутов

ABAC



# На основе списка

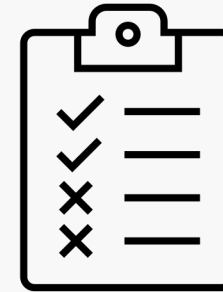
ACL



Субъект



Объект



Список

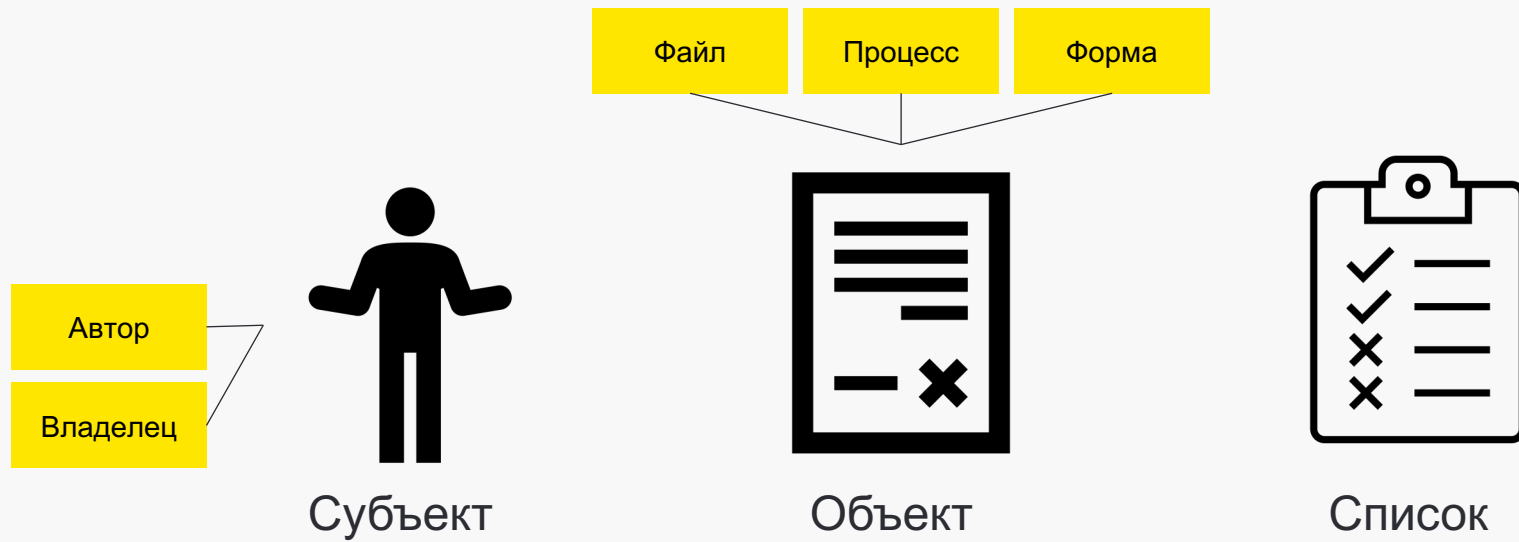
# На основе списка

ACL



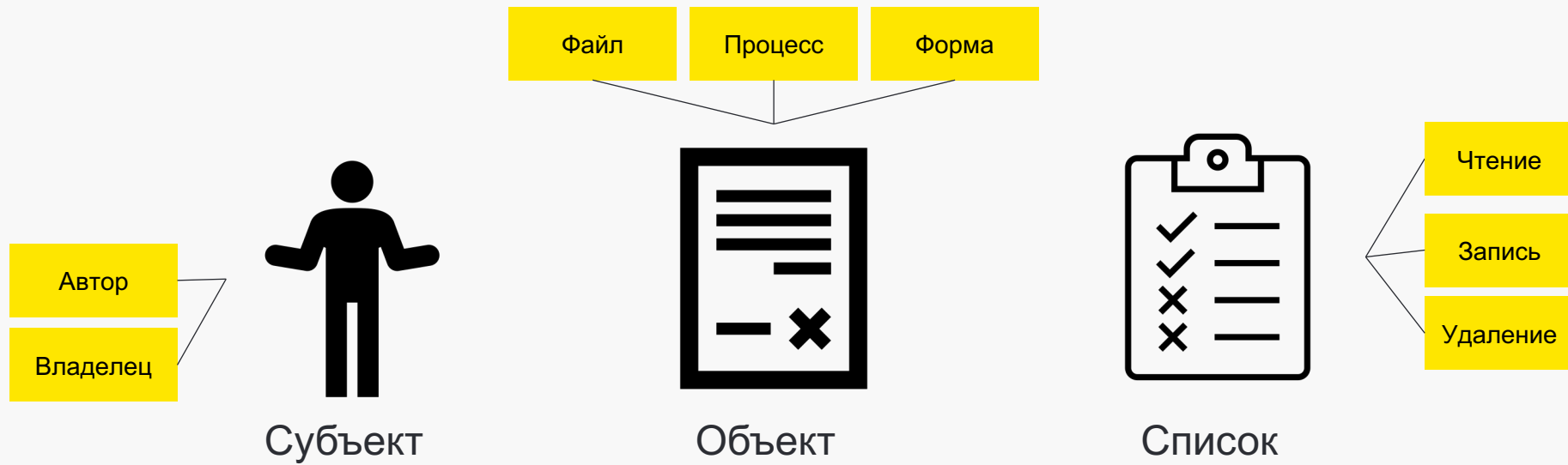
# На основе списка

ACL



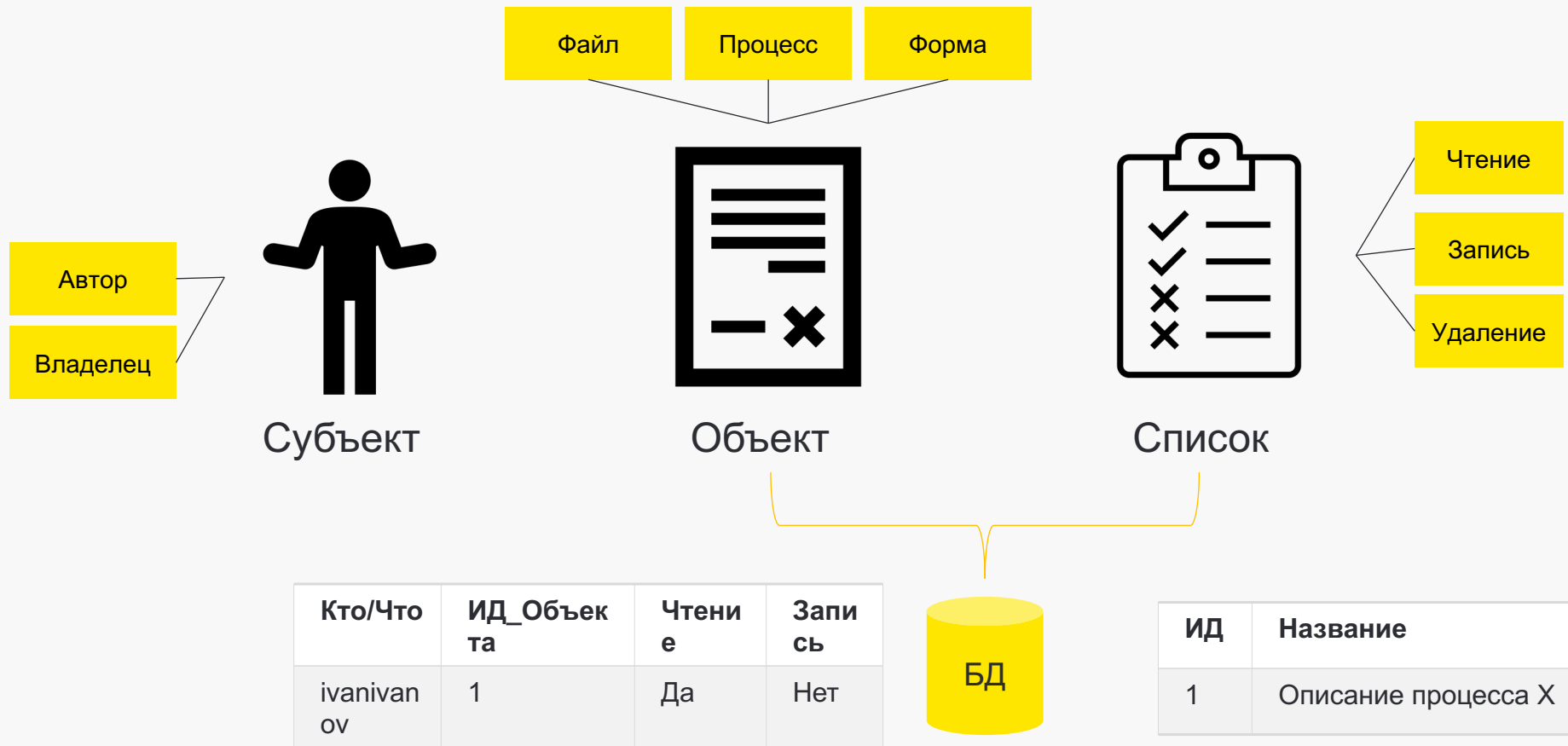
# На основе списка

ACL



# На основе списка

ACL



# На основе политик

RBAC + ABAC + ACL

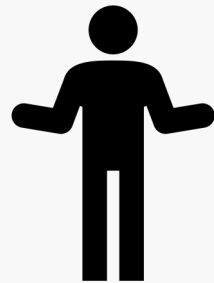
Модели

RBAC

ABAC

ACL

Политики



Субъект



Объект

Политика



Список



Роли



Атрибуты



# На основе политик

RBAC + ABAC + ACL

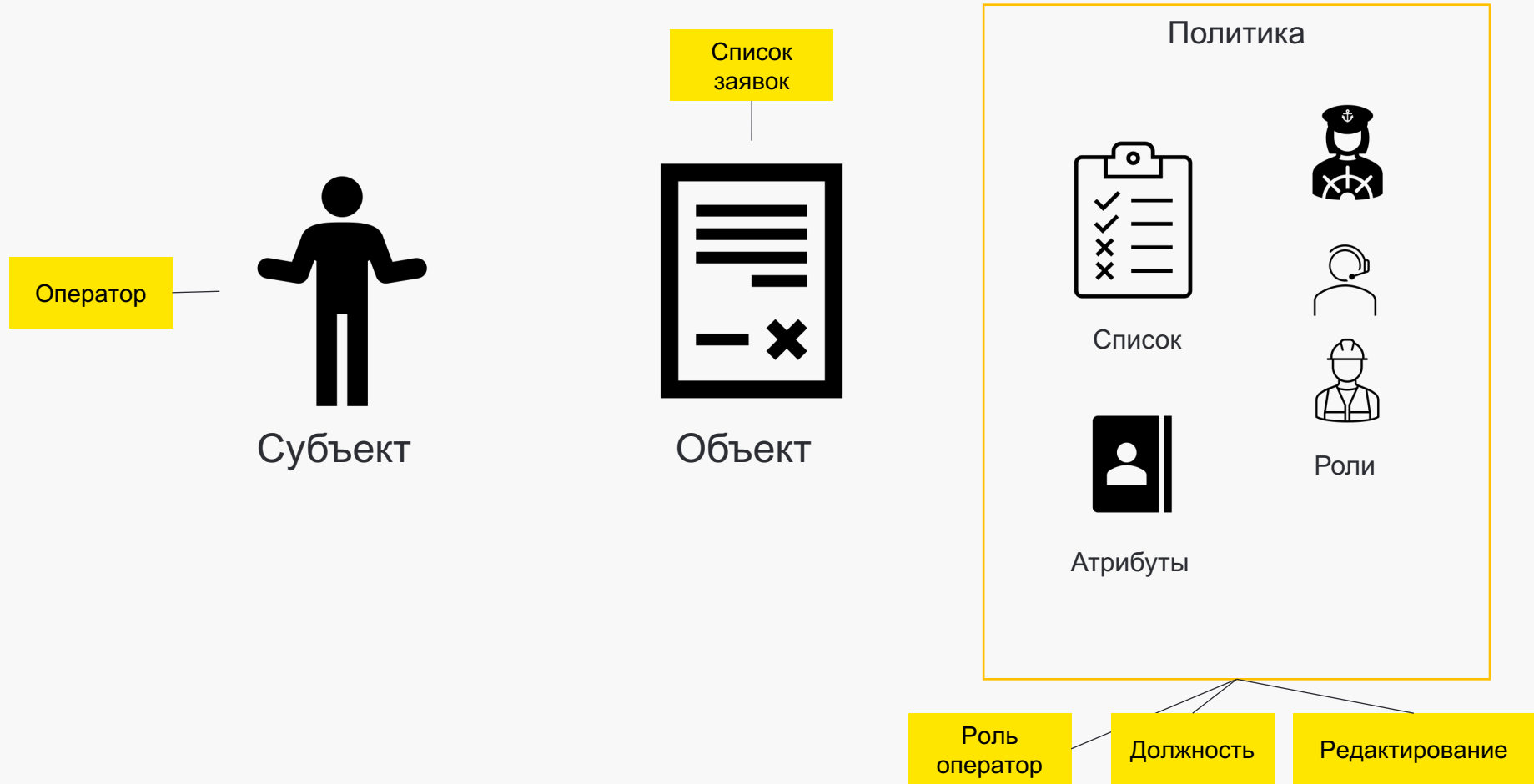
Модели

RBAC

ABAC

ACL

Политики



# Авторизация с использованием LDAP и Kerberos



# Простая авторизация

Negotiate - Kerberos

```
builder.Services
```

```
    .AddAuthorization()
```

```
    .AddAuthentication(NegotiateDefaults.AuthenticationScheme)
```

```
    .AddNegotiate();
```

```
app.UseAuthentication();
```

```
app.UseAuthorization();
```

# Простая авторизация

Negotiate - Kerberos

```
app.MapGet("RequireAuthorization", async context =>
{
    // Обработка запроса.
})
.RequireAuthorization(); // Требуем авторизации для этой конечной точки.

{
    "Title": "Привет! Эта конечная точка не требует авторизации,
а значит и аутентификацию не выполняет.",
    "Link": {
        "rel": "RequireAuthorization",
        "description": "Перейдите по ссылке, чтобы проверить
Negotiate аутентификацию.",
        "href":
"https://examples.raiffeisen.ru:7282/RequireAuthorization"
    },
    "IsAuthenticated": false,
    "Name": null,
    "Claims": []
}
```

# RBAC

Negotiate – Kerberos + LDAP

```
builder.Services.AddAuthentication(NegotiateDefaults.AuthenticationScheme)
    .AddNegotiate(options =>
    {
        options.EnableLdap(settings =>
        {
            builder.Configuration.GetSection("Ldap").Bind(settings);

            // Порт указывать обязательно. Он не разрешается автоматически как в протоколе HTTP.
            var di = new LdapDirectoryIdentifier(settings.Domain, 636, true, false);

            // если вы явно указываете имя и пароль, удалите if или реализуйте else.
            if (string.IsNullOrEmpty(settings.MachineAccountName))
            {
                settings.LdapConnection = new LdapConnection(di)
                {
                    SessionOptions = { ProtocolVersion = 3, SecureSocketLayer = true }
                };
            }
        });
    });
```

# RBAC

Negotiate – Kerberos + LDAP

```
{  
  "Title": "Привет! Эта конечная точка не требует авторизации, а значит и  
аутентификацию не выполняет.",  
  "Link": {  
    "rel": "RequireAuthorization",  
    "description": "Перейдите по ссылке, чтобы проверить Negotiate  
аутентификацию.",  
    "href": "https://examples.raiffeisen.ru:7282/RequireAuthorization"  
  },  
  "IsAuthenticated": false,  
  "Name": null,  
  "Claims": [  
    {"role": "RBAC-User"},  
    {"role": "RBAC-SF-User"},  
    {"role": "RBAC-XYZ-Admin"}  
  ]  
}
```

# ABAC

Negotiate – Kerberos + LDAP

```
builder.Services.AddAuthentication(NegotiateDefaults.AuthenticationScheme)
    .AddNegotiate(options =>
    {
        options.Events = new NegotiateEvents()
        {
            OnRetrieveLdapClaims = async (context) =>
            {
                // Работает с context.LdapSettings.LdapConnection
                // И др. методами из System.DirectoryServices.Protocols
            }
        };
    });
```

## Negotiate – Kerberos + LDAP

```
{  
  "Title": "Привет! Эта конечная точка не требует авторизации,  
а значит и аутентификацию не выполняет.",  
  "Link": {  
    "rel": "RequireAuthorization",  
    "description": "Перейдите по ссылке, чтобы проверить  
Negotiate аутентификацию.",  
    "href":  
      "https://examples.raiffeisen.ru:7282/RequireAuthorization"  
  },  
  "IsAuthenticated": false,  
  "Name": null,  
  "Claims": [  
    {"role": "RBAC-User"},  
    {"role": "RBAC-SF-User"},  
    {"role": "RBAC-XYZ-Admin"},  
    {"position": "Senior Community Lead"},  
    ...  
  ]  
}
```



# Вызов из HttpClient

Negotiate – Kerberos + LDAP

```
var client = new HttpClient(  
    new HttpClientHandler  
    {  
        Credentials = new NetworkCredential(Username, UserPassword)  
    });
```

```
var response = await client.GetAsync(ProfilesUrl);
```

```
var client = new HttpClient(  
    new HttpClientHandler  
    {  
        UseDefaultCredentials = true  
    });
```




```
var response = await client.GetAsync(ProfilesUrl);
```



# Авторизация с использованием Keycloak



# Identity and Access Management

-  Single-Sign On  
User Federation: LDAP, AD и другие хранилища  
Брокер Kerberos
-  Открытый исходный код  
Стандартные протоколы: OpenID Connect, O'Auth 2.0, SAML
-  Имеет сертификат соответствия по ряду профилей  
FAPI, AU-CDR, BR-OB.



# Где найти?

На официальном сайте [www.keycloak.org](http://www.keycloak.org).

- **Спонсор проекта:** Red Hat
- **Лицензия:** Apache 2.0
- **Сертифицировано:** Red Hat

Подробнее о сертификации на OpenID Foundation и альтернативах keycloak смотрите на сайте [www.openid.net](http://www.openid.net)





**.NET**  
Community

# Live code



# Итоги



# Полезное

1. [Репозиторий сообщества с примерами](#)
2. [Запись доклада на DotNetRu meetup с подробным описанием Negotiate](#)

Репозиторий с примерами





# Спасибо



E-mail: [Prosin.Roma@ya.ru](mailto:Prosin.Roma@ya.ru)

Telegram: @ProsinRoman