Utilizando Modelagem de Ameaças em uma página de Login

Who we are...

Tiago Zaniquelli

- Pai da Bia;
- Desde 2003 atuando com desenvolvimento;
- Desde 2021 atuando com AppSec na Conviso Application Security;
- Apaixonado por Segurança, Agilidade e Desenvolvimento;

Redes

- Twitter: zani0x03;
- Linkedin: https://br.linkedin.com/in/tiago-zaniquelli
- GitHub: https://github.com/zani0x03

E-mail

- zani0x03 'at' gmail 'dot' com
- tiago.zaniquelli 'at' owasp 'dot' org

Visão Geral

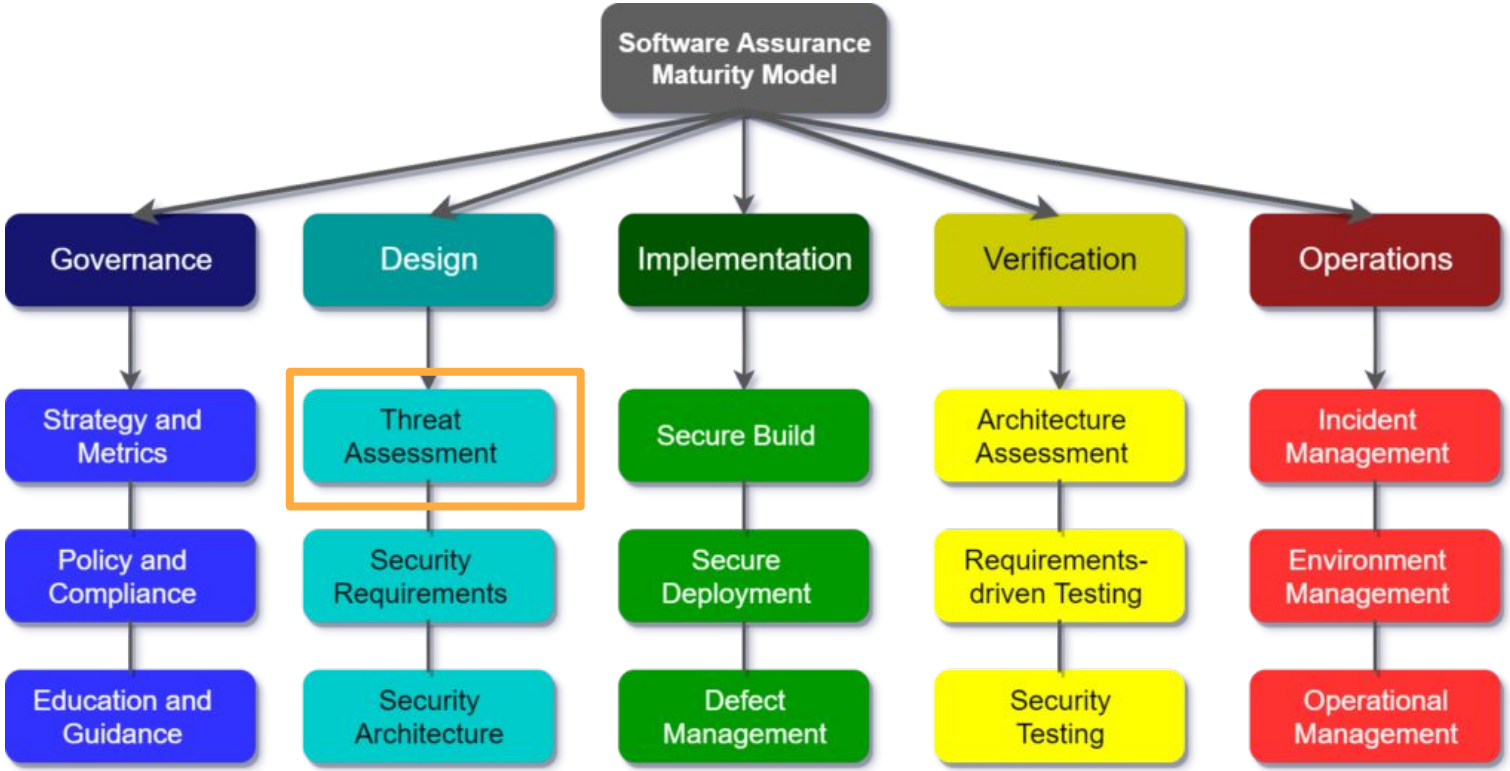
- Modelagem de ameaças;
- Modelagem de ameaças SAMM;
- CAPEC;
- CWE;
- ASVS;
- Exemplo;

Modelagem de ameaças

O que é modelagem de ameaças?

É uma **técnica efetiva** que ajuda a construir aplicações, sistemas, redes e serviços de **maneira segura**. De forma que **identifica ameaças potenciais** e **reduz riscos** estratégicos logo no início do ciclo de desenvolvimento.

OWASP SAMM


Software Assurance Maturity Model
Governance
Design
Implementation
Verification
Operations
Strategy and Metrics
Policy and Compliance
Education and Guidance
Threat Assessment
Security Requirements
Security Architecture
Secure Build
Secure Deployment
Defect Management
Architecture Assessment
Requirements-driven Testing
Security Testing
Incident Management
Environment Management
Operational Management

Modelagem de Ameaças dentro das práticas do OWASP SAMM Nível 1

Você identifica e gerencia falhas de projeto de arquitetura com modelagem de ameaças?

- Você **executa a modelagem de ameaças** para aplicativos de alto risco;
- Você usa listas de verificação de ameaças simples, como STRIDE;
- Você persiste o resultado de um modelo de ameaça para uso posterior.

Modelagem de Ameaças dentro das práticas do OWASP SAMM Nível 2

Você usa uma **metodologia padrão**, alinhada aos níveis de risco de sua aplicação?

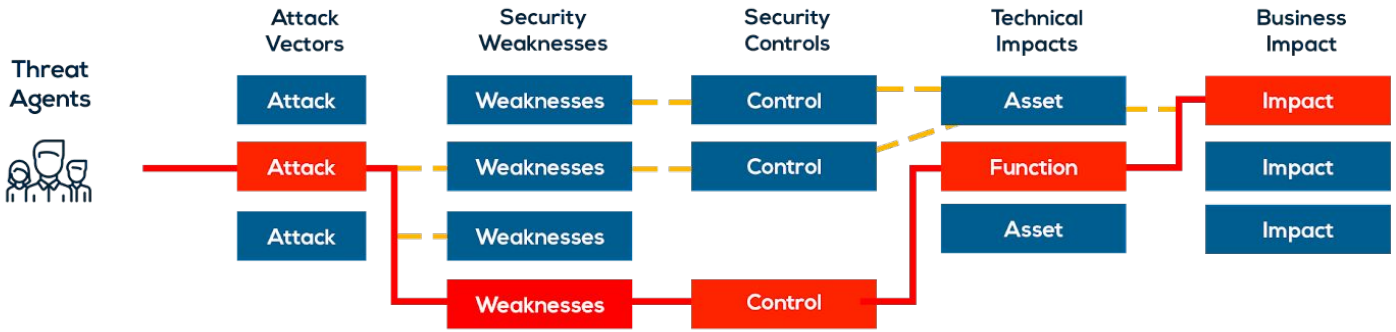
- Você treina seus arquitetos, security champions e outras partes interessadas sobre como fazer modelagem de ameaças na prática?
- Sua metodologia de modelagem de ameaças inclui pelo menos diagramação, identificação de ameaças, mitigação de falhas de design e como validar seus artefatos de modelo de ameaças?
- Mudanças no aplicativo ou contexto de negócios acionam uma revisão dos modelos de ameaças relevantes?

Modelagem de Ameaças dentro das práticas do OWASP SAMM Nível 3

Você revisa e atualiza regularmente a metodologia de modelagem de ameaças para seus aplicativos?

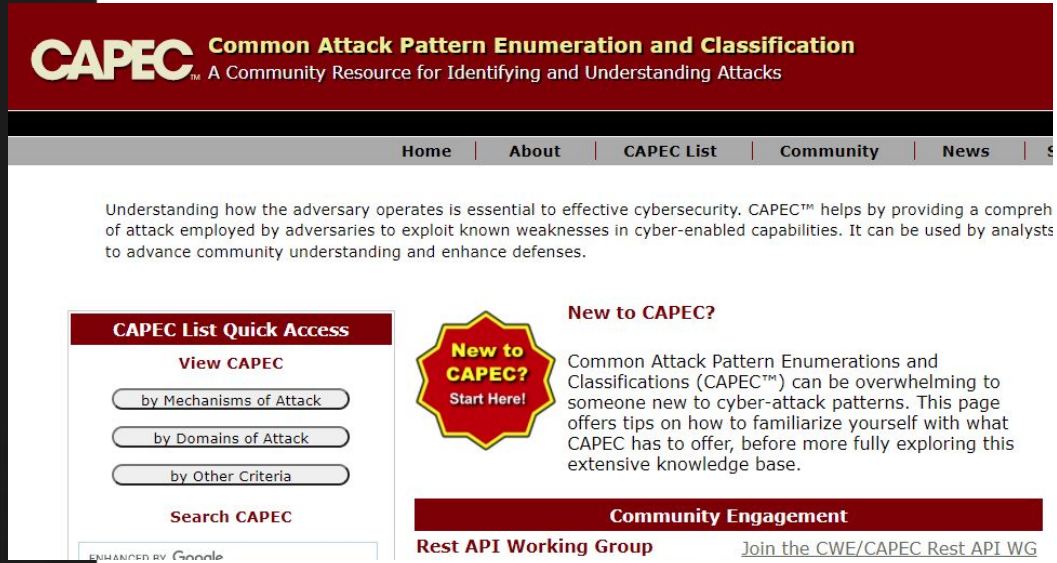
- A metodologia do modelo de ameaça considera o feedback histórico para melhoria?
- Você regularmente (por exemplo, anualmente) revisa os modelos de ameaças existentes para verificar se nenhuma nova ameaça é relevante para seus aplicativos?
- Você automatiza partes do seu processo de modelagem de ameaças com ferramentas de modelagem de ameaças?

Por que realizar Modelagem de ameaças?



CAPEC™

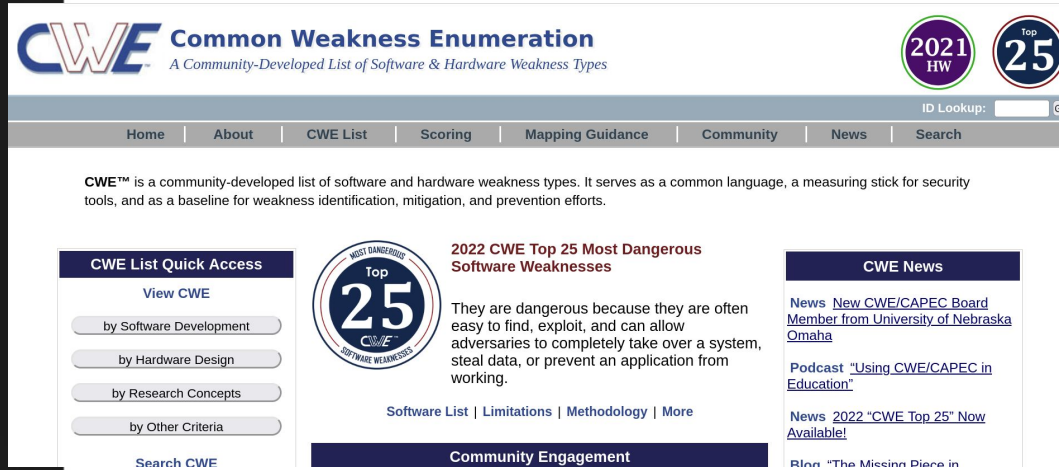
O **Common Attack Pattern Enumeration and Classification** (CAPEC™) fornece um catálogo público de **padrões de ataque comuns** que ajuda os usuários a entender como os adversários exploram pontos fracos em aplicativos e outros recursos cibernéticos.



CWE™

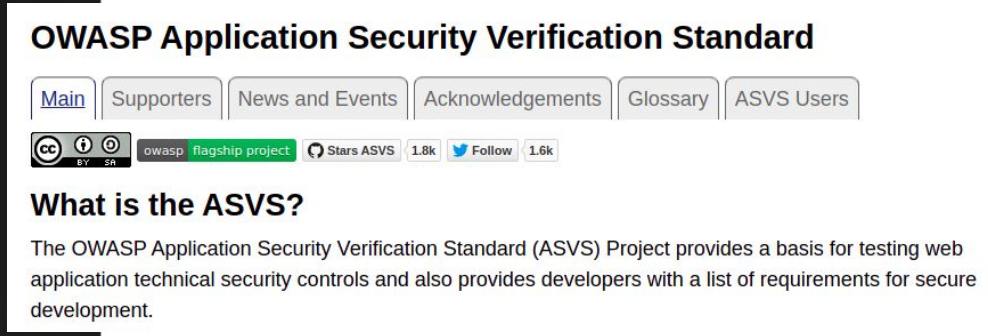
Common Weakness Enumeration (CWE™) é uma lista de tipos de fraquezas de software e hardware, desenvolvida pela comunidade.

Ela serve como uma linguagem comum, uma medida para ferramentas de segurança e como uma linha de base para os esforços de **identificação, mitigação e prevenção de fraquezas.**



ASVS

O projeto OWASP Application Security Verification Standard (**ASVS**), provê uma base para testar **controles básicos** em suas aplicações web e também provê para os desenvolvedores uma **lista de requisitos de segurança** para desenvolvimento seguro.



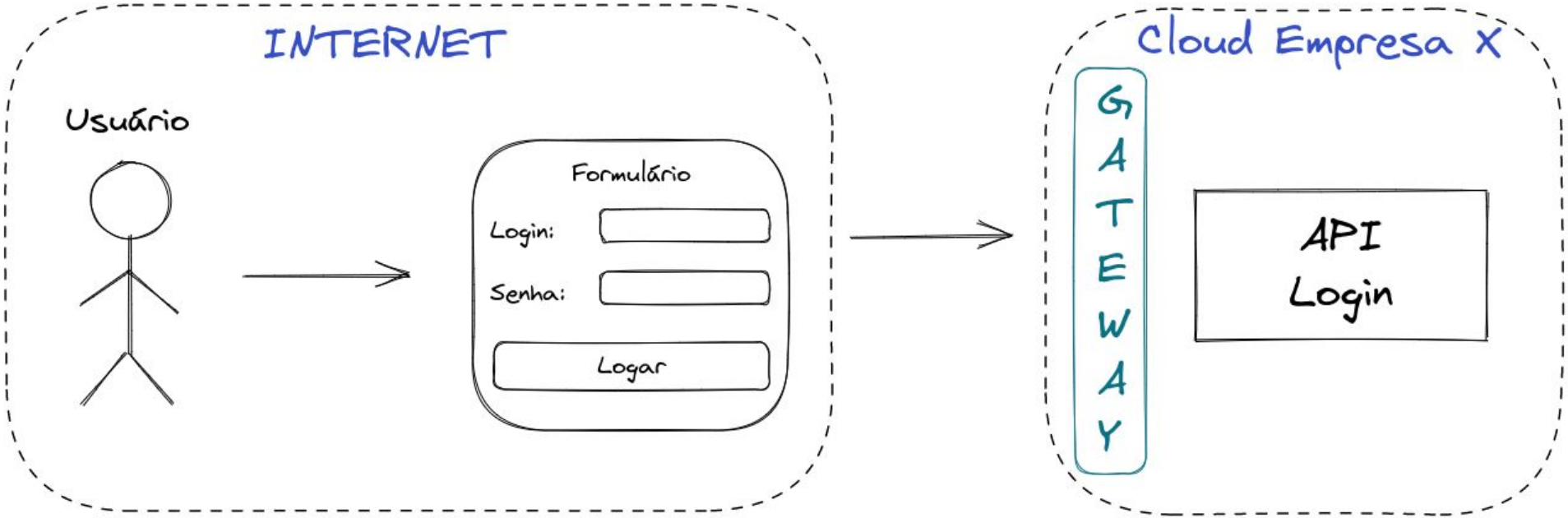
Testes orientados a requisitos

SAMM model overview

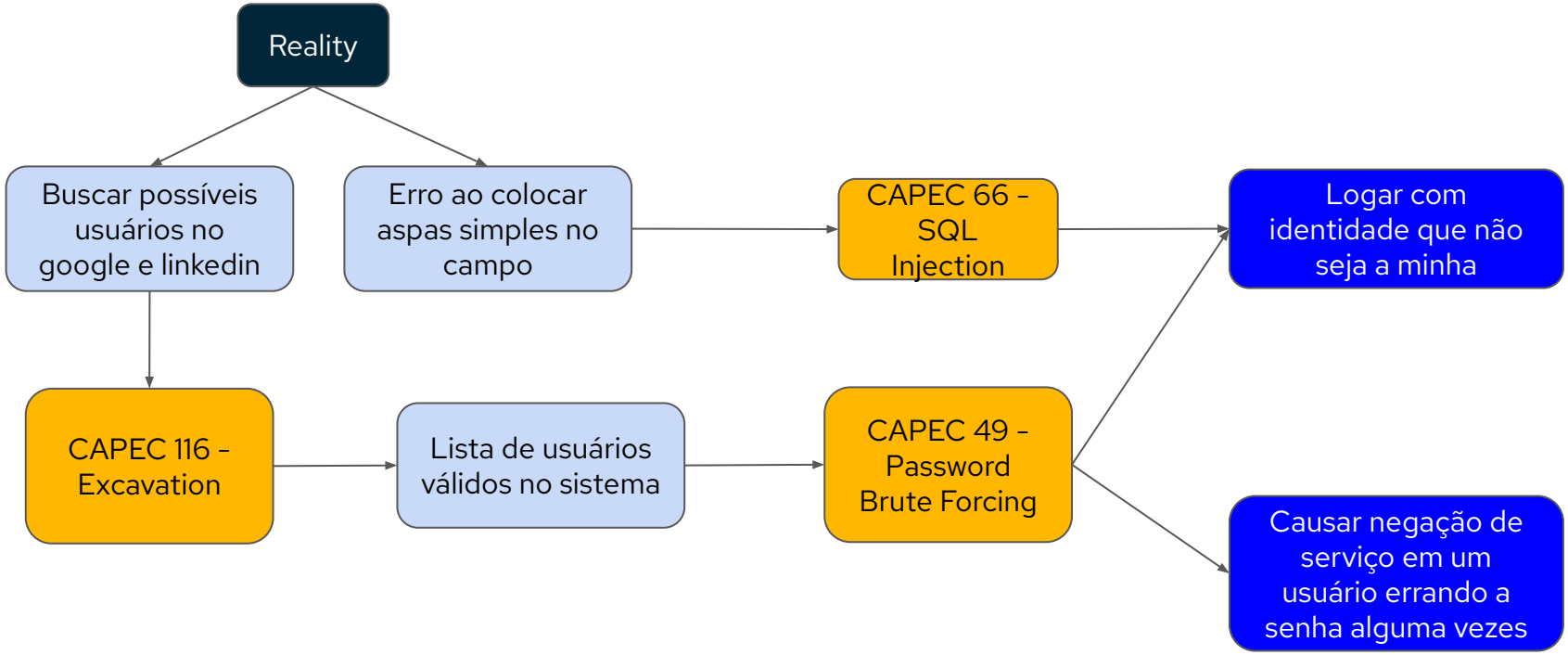
Governance	Design	Implementation	Verification	Operations
Strategy and Metrics	Threat Assessment	Secure Build	Architecture Assessment	Incident Management
Policy and Compliance	Security Requirements	Secure Deployment	Requirements-driven Testing	Environment Management
Education and Guidance	Security Architecture	Defect Management	Security Testing	Operational Management

Exemplo

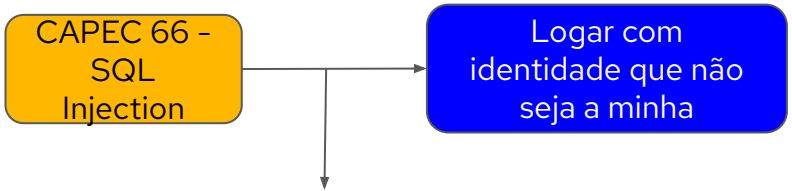
Cenário


INTERNET
Usuário
Formulário
Login:
Senha:
Logar
Cloud Empresa X
GATEWAY
API
Login

Modelagem de Ameaças - Atacante



Modelagem de Ameaças – Defesa



Related Weaknesses

CWE-ID	Weakness Name
89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
1286	Improper Validation of Syntactic Correctness of Input

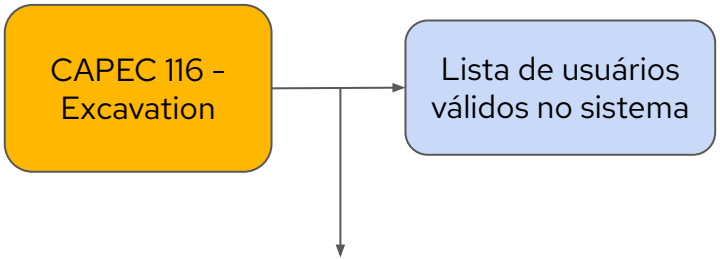
#	Description	L1	L2	L3	CWE
5.3.5	Verify that where parameterized or safer mechanisms are not present, context-specific output encoding is used to protect against injection attacks, such as the use of SQL escaping to protect against SQL injection. (C3, C4)	✓	✓	✓	89

Código Vulnerável e Código Corrigido

```
1 reference
public static async Task<User> LoginSQL(LoginRequest login)
{
    var conn = SqliteConfigConnection.GetSQLiteConnection();
    string query = "Select id, name, login, password, dateInsert, dateUpdate, isAdmin, inativo, dateChangePassword from users "  +
        "where login = '"+login.Login+"' and password = '"+UtilService.ReturnSha512(login.Password)+"' and inativo = 0";
    var user = await conn.QueryAsync<User>(query);
    return user.FirstOrDefault();
}
```

```
public static async Task<User> LoginSQL(LoginRequest login)
{
    var conn = SqliteConfigConnection.GetSQLiteConnection();
    string query = "Select id, name, login, password, dateInsert, dateUpdate, isAdmin, inativo," +
        "dateChangePassword from users where login = @login and password = @password and inativo = 0";
    var user = await conn.QueryAsync<User>(query, new{
        @login = login.Login,
        @password = UtilService.ReturnSha512(login.Password)
    });
    return user.FirstOrDefault();
}
```

Modelagem de Ameaças – Defesa



Related Weaknesses

CWE-ID	Weakness Name
200	Exposure of Sensitive Information to an Unauthorized Actor
1243	Sensitive Non-Volatile Information Not Protected During Debug

Authentication Responses

Using any of the authentication mechanisms (login, password reset or password recovery), an application must respond with a generic error message regardless of whether:

Código Vulnerável e Código Corrigido

```
[HttpPost]
[Route("login")]
0 references
public async Task<ActionResult> Login([FromBody]LoginRequest login)
{
    // Recupera o usuário
    var user = await UserRepository.Login(login);


    // Verifica se o usuário existe
    if (user == null)
    {
        return Unauthorized(new
        {
            message = "User not found!"
        });
    }

    if (user.Password == UtilService.ReturnSha512(login.Password)){
        if (login.IsAdmin.HasValue)
            user.IsAdmin = login.IsAdmin.Value;
        user.Password = "";
        var token = TokenService.GenerateToken(user);
        return Ok(new
        {
            User = user,
            token = token
        });

    }
    else{
        return Unauthorized(new
        {
            message = "Wrong password!!!"
        });
    }
}
```

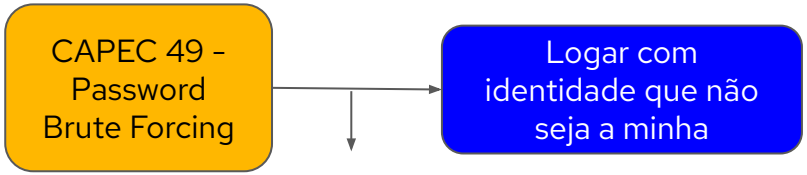
```
[HttpPost]
[Route("login")]
0 references
public async Task<ActionResult> Login([FromBody]LoginRequest login)
{

    try{
        // Recupera o usuário
        var user = await UserRepository.Login(login);


        // Verifica se o usuário existe
        if (user == null)
        {
            return Unauthorized(new
            {
                message = "User/Password wrong!!"
            });
        }

        user.Password = "";
        var token = TokenService.GenerateToken(user);
        return Ok(new
        {
            User = user,
            token = token
        });
    }catch(Exception ex){
        _logger.LogError(ex, "General error");
        return StatusCode(500, "Internal server error");
    }
}
```

Modelagem de Ameaças – Defesa



Related Weaknesses

CWE-ID	Weakness Name
521	Weak Password Requirements
262	Not Using Password Aging
263	Password Aging with Long Expiration
257	Storing Passwords in a Recoverable Format
654	Reliance on a Single Factor in a Security Decision
307	Improper Restriction of Excessive Authentication Attempts
308	Use of Single-factor Authentication
309	Use of Password System for Primary Authentication

#	Description	L1	L2	L3	CWE	NIST §
2.2.1	Verify that anti-automation controls are effective at mitigating breached credential testing, brute force, and account lockout attacks. Such controls include blocking the most common breached passwords, soft lockouts, rate limiting, CAPTCHA, ever increasing delays between attempts, IP address restrictions, or risk-based restrictions such as location, first login on a device, recent attempts to unlock the account, or similar. Verify that no more than 100 failed attempts per hour is possible on a single account.	✓	✓	✓	307	5.2.2 / 5.1.1.2 / 5.1.4.2 / 5.1.5.2

Código Corrigido

```
builder.Services.Configure<IpRateLimitOptions>(options =>
{
    options.EnableEndpointRateLimiting = true;
    options.StackBlockedRequests = false;
    options.HttpStatusCode = 429;
    options.RealIpHeader = "X-Real-IP";
    options.GeneralRules = new List<RateLimitRule>
        {
            new RateLimitRule
            {
                //Endpoint = "*",
                Endpoint = "POST:/api/User/passwordrecovery",
                Period = "60s",
                Limit = 2
            },
            new RateLimitRule
            {
                Endpoint = "POST:/api/authentication/login",
                Period = "60s",
                Limit = 3
            }

        };
});
```

Obrigado!!!

Referências

OWASP

https://owasp.org/www-project-top-ten/
https://owasp.org/www-project-application-security-verification-standard/
https://owaspsamm.org/
https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

Conviso (2022)

https://www.convisoappsec.com/

Mitre (2022)

https://cwe.mitre.org/
https://capec.mitre.org/

Common Credentials

https://github.com/danielmiessler/SecLists/blob/master/Passwords/Common-Credentials/top-passwords-shortlist.txt