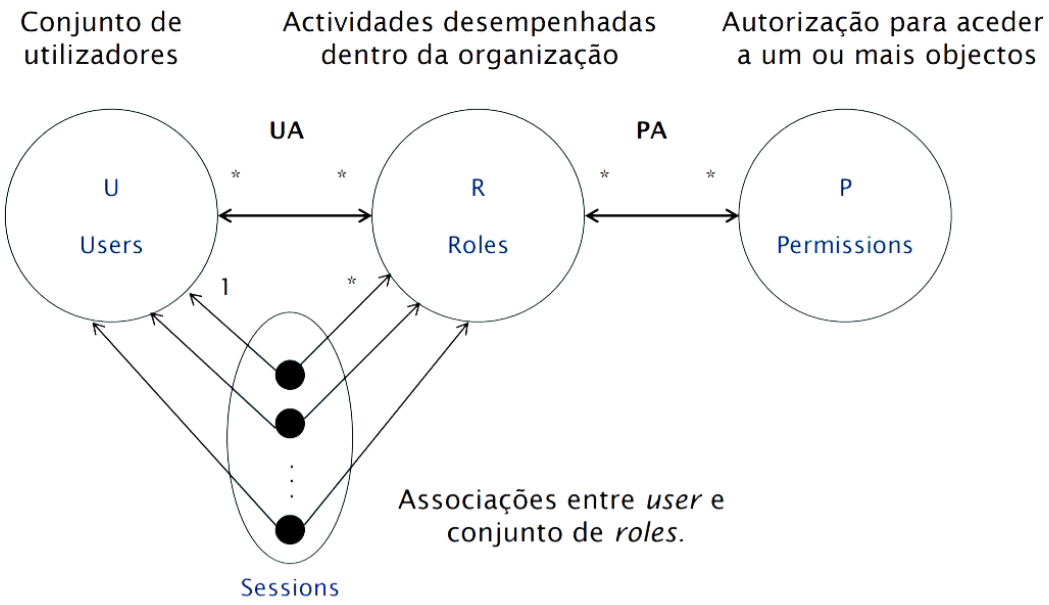
Segurança Informática

Licenciatura em Engenharia Informática, Redes e Telecomunicações

3ª Série de Exercícios



Grupo 4

43874 João Florentino

46435 Mihail Ababii

46919 Bárbara Castro

1º Semestre - Ano letivo 2020/2021

24 de janeiro de 2021

Resumo

Este trabalho incide sobre políticas, modelos e mecanismos de controlo de acessos; Matriz de controlo de acessos, as suas capacidades, lista de controlo de acessos e alguns casos práticos; Conjuntos e relações RBAC0, RBAC1, RBAC2, RBAC3; Segurança no desenvolvimento de *software;* Vulnerabilidades de *buffer overflow;* Vulnerabilidades em aplicações web; Ataques do tipo *Cross-Site Request Forgery* (CSRF) e *Cross-Site Scripting*(XSS)

Palavras-chave: aplicação *web*; ataque; autenticação; *browser*; *buffer*; *callback*; cifra; *cookies;* *Cross-Site Request Forgery*; *Cross-Site Scripting*; modelo $RBAC_1$; modelo $RBAC_2$; modelo $RBAC_3$; S*ame-Origin Policy*; privilégios; *scripts*; URL; vulnerabilidades; *query*

Índice

Lista de Figuras

1. Introdução

Na realização deste trabalho, foi-nos proposto que estudássemos o protocolo OpenID Connect com a *framework* OAuth2.

Numa primeira parte foi-nos proposto que estudássemos quais os elementos que foram adicionados ao OAuth 2.0 com a vinda do OpenID Connect. De seguida foi-nos pedido que estudássemos em termo de ACLs qual o tipo de protocolo que a framework OAuth 2.0 se enquadrava.

Numa segunda fase foi-nos pedido que estudássemos os níveis de RBAC que suportavam Least Privilege e Separation of Duty. Foi-nos ainda pedido que através de uma biblioteca de autorização, com o nome de Casbin, implementássemos o modelo de segurança descrito no enunciado do trabalho.

Por último estudamos uma vulnerabilidade real chamada *buffered overflow* que pega numa vulnerabilidade do código c/c++ para rescrever código malicioso dentro de aplicações. Ainda neste âmbito de vulnerabilidades realizamos testes a um site chamado Google Gruyere que apresenta uma vulnerabilidade, vulnerabilidade essa que permite realizar um ataque de Cross-Site Request Forgery (CSRF) e um ataque de Cross-Site Scripting (XSS).

Ferramentas utilizadas para o desenvolvimento do trabalho

No desenvolvimento do trabalho aqui descrito, recorremos à utilização da linguagem de programação JavaScript e HTML para realizar os programas que se encontram descritos nas alíneas 6.

2. Resolução dos problemas do trabalho

Problema 1:

No contexto do protocolo OpenID Connect:

Que elementos foram adicionados em relação à framework OAuth 2.0?

O OpenID Connect acrescenta à framework OAuth 2.0 uma camada de identidade. Esta camada é conseguida através da autenticação do utilizador através de um conjunto assinado de asserções (id_token). Tambem é possível aceder a informação adicional sobre o utilizador indicando o access token.

Se a aplicação cliente (relying party) usar 2 fornecedores de identidade, e tiver um total de 100 utilizadores, quantos client_id tem de gerir?

A aplicação cliente terá de ter 1 client_id por cada fornecedor de identidade, pelo que ao ter 2 fornecedores de identidade a aplicação cliente terá 2 client_id. O número de cliente_id's não é dependente do número de utilizadores que a aplicação tiver.

Problema 2:

Tendo em conta as abordagens de controlo de acessos estudadas, lista de controlo de acessos (ACL) e lista de capacidades, indique em que tipo de controlo se pode enquadrar a estrutura access_token da framework OAuth 2.0.

Apos o login é atribuído ao utilizador um access token onde estão presentes security identifiers (SID) com a identificação do utilizador e dos grupos a que pertence. Apos a criação do recurso é lhe associado um security deacriptor com o DACL (Discretionary Access Control List). Devido este, a estrutura access_tokenpode ser enquadrada neste tipo de controlo de accessos através de DACL.

Problema 3:

Considere os diferentes níveis do modelo RBAC.

De que forma são suportados os princípios de segurança Least Privilege e Separation Of Duty?

O princípio de segurança Least Privilege é suportado e adicionado ao modelo RBAC através de RBAC1 que serve para limitar os roles herdados. Um utilizador escolhe qual o role que quer ativa, herdando os roles júnior desse. O princípio de segurança Separation of Duty, é adicionado através de RBAC2 que adiciona suporte para *duty policies*, ou seja, serve para impor regras na organização. Têm a forma de predicados, retornando "aceite" ou "não aceite".

É possível existir uma sessão associada ao utilizador u e com o role r activo, sem que (u, r) esteja na relação user assignment (UA)?

Sim, é possível existir, pois pode existir uma hierarquia de roles na qual r apresenta um sénior, sendo que este esta associado ao utilizador u na relação user assignment (u, r_{senior}).

Problema 4:

Considere a biblioteca de autorização Casbin [1]:

Explique de que forma uma aplicação consegue através desta biblioteca configurar um modelo de segurança e a concretização de uma política.

Esta biblioteca consegue configurar um modelo de segurança e a concretização de uma política tendo um modelo de controlo de acesso baseado em PERM metamodel, ou seja, um moledo abstrato que permite adaptar a configuração das políticas (P), dos efeitos (E), dos pedidos (R), e dos respetivos resultados (M) de acordo com o modelo pretendido. Deste modo é possível, relacionar os diferentes modelos, partilhando um mesmo conjunto de políticas, ou adaptar os modelos existentes.

Considere a seguinte política definida usando o modelo RBAC1:

- **U = {u1, u2}, R = {r0, r1, r2, r3}, P = {p0, p2, p3}**
- **{r0 ≤ r1, r0 ≤ r2, r2 ≤ r3} ⊆ RH**
- **UA = {(u1, r1), (u2, r2), (u3, r3)}**
- **P A = {(r0, p0), (r2, p2), (r3, p3)}**

request		**policy**
Accept	Denny	
		p, r3, file, p3
u3, file, p3	u1, file, p3	p, r2, file, p2
u3, file, p2	u1, file, p2	p, r0, file, p0
u3, file, p0	u2, file, p3	
u2, file, p2		g, u3, r3
u2, file, p0		g, u2, r2
u1, file, p0		g, u1, r1
		g, r3, r2
		g, r2, r0
		g, r1, r0

Problema 5:

Descreva sucintamente os seguintes aspetos sobre a CVE-2020-8962 [4].

Qual o tipo de vulnerabilidade

Vulnerabilidade do tipo *stack-based buffer overflow.*

Que tipo de software é alvo do ataque

As aplicações escritas em c/c++ são vulneráveis a este tipo de ataque uma vez que podem realizar aritmética de endereços e aceder arbitrariamente à memoria. No caso particular exposto, o alvo do ataque foi um router da D-link com a seguinte referencia *D-Link DIR-842 REVC*

Como pode a vulnerabilidade ser explorada

A vulnerabilidade consiste em escrever código malicioso fora do buffer de memoria ou meter mais dados do que este consiga aguentar. Deste modo a informação já existente no buffer é rescrita, inclusive o ponto de retorno, ou seja, da próxima vez que o programa executar, irá executar o ponteiro para o código malicioso.

Problema 6:

Inicie uma instância da aplicação e indique o id no relatório da série.

Mihail: Your Gruyere instance id is **498260413190975665359340215529512337116**.
Joao: Your Gruyere instance id is **354189677610052710631975174902398196525.**

Realize o desafio de Cross-site Request Forgery (CSRF) descrevendo como configurou a aplicação de ataque. Apresente também um esquema/diagrama com a solução proposta para resolver a vulnerabilidade na aplicação Gruyere.

Começamos por criar um servidor com o modulo express. De seguida configuramos duas rotas em que uma era "ativada" quando era clicado no link apresentado em html e a outra era ativada ao tentar importar uma imagem. Em ambas o site para quem estas rotas iam fazer o pedido era para gruyère de forma a assim apagar o snippet criado pelo utilizador.

```
app.use("/image", (req, res) => {
    res.send(`<img src="https://google-gruyere.appspot.com/354189677610052710631975174902398196525/deletesnippet?index=0">`);
})
app.use("/", (req, res) => {
    res.send(`<h1><a href="https://google-gruyere.appspot.com/354189677610052710631975174902398196525/deletesnippet?
    index=0">!!!Click here!!!</a>`);
})
```

Figura 1-Rotas para pedidos

A forma de resolver este problema seria mudando o tipo do método request, ou seja, em vez de realizar o pedido com um método GET poderia realizar com método POST que assim iria alterar o estado da página. Outra forma de resolver o problema seria enviar um authorization token para o cliente e verificá-lo quando este realizasse um pedido para o site Gruyere. A última forma, que foi implementada no google Chrome, seria evitar que fossem enviadas cookies de um site para outro.

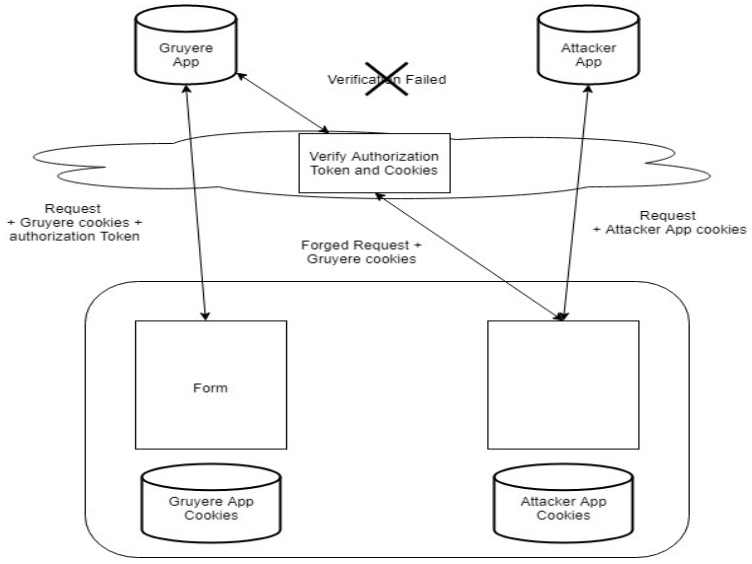


Figura 2- Solução para CSRF

3. Conclusão

Na realização desta série de exercícios, estudámos políticas, modelos e mecanismos de controlo de acessos; matriz de controlo de acessos, nomeadamente as suas capacidades, lista de controlo de acessos e alguns casos práticos; modelos RBAC; conjuntos e relações $RBAC_0$ a $RBAC_3$; segurança no desenvolvimento de *software*; vulnerabilidades de *buffer overflow;* vulnerabilidades em aplicações *web*; segurança no *browser*, nomeadamente a política da mesma origem; ataques do tipo *Cross-Site Request Forgery* (CSRF) e *Cross-Site Scripting* (XSS).

A realização deste trabalho permitiu-nos contactar diretamente com aplicações *Web* e, através desse contacto, aprofundámos conhecimentos acerca das vulnerabilidades a nível de desenvolvimento de *software* e das medidas de segurança que devemos colocar em prática de forma a evitar determinados tipos de ataques comos os referidos anteriormente.

Utilizámos a linguagem de programação *Javascript* e HTML para criarmos os programas necessários e realizarmos vários cenários de teste.

4. Referências:

- Casbin - https://casbin.org/
- RBAC: Hierarchical Role Based Access Control -https://www.npmjs.com/package/rbac
- Casbin RBAC model - https://casbin.org/docs/en/rbac
- CVE-2020-8962 - https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-8962
- Google Gruyere - Web Application Exploits and Defenses - https://googlegruyere.appspot.com
- Google Gruyere - start new instance - https://google-gruyere.appspot.com/start
- Google Gruyere - Cross-site Scripting - https://google-gruyere.appspot.com/part2#2__xss_challenge
- Google Gruyere - Cross-site Request Forgery - https://google-gruyere.appspot.com/part3#3__cross_site_request_forgery