Instituto Superior de Engenharia de Lisboa

Licenciatura/Mestrado em Engenharia Informática e de Computadores

Segurança Informática

Segunda série de exercícios, Semestre de Inverno de 08/09

Data de entrega: 29 de Novembro de 2011

---

1. Considere o artigo de Halderman et al. [1]:

   1.1. Qual o problema tratado neste artigo?

   1.2. O que é um "phishing attack"? De que forma a proposta apresentada neste artigo oferece protecção contra este tipo de ataque? Descreva sucintamente outros métodos propostos para evitar este tipo de ataque.

   1.3. A informação inserida pelo utilizador para a derivação das "passwords" inclui um "username". Qual a finalidade desta componente?

   1.4. Justifique a divisão do processo de derivação de "passwords" específicas em duas fases.

   1.5. Quais os tipos de ataque considerados? Justifique esta classificação.

2. Um programa é designado por "Cavalo de Tróia" se, parecendo cumprir um determinado objectivo legítimo, realiza outras acções não legítimas e que comprometem a integridade do sistema. Em que medida o controlo de acessos mandatório (MAC) impede esta ameaça? Use o modelo *Biba* como referência.

3. Pretende-se implementar uma política de controlo de acesso, baseada no modelo de Bell-LaPadula, sobre um mecanismo que suporta o modelo $RBAC_1$. A política de Bell-LaPadula é caracterizada por:

   - Reticulado constituído pelo seguinte conjunto de etiquetas $\{l_0, l_1, l_2, l_3\}$, onde $l_0 \leq l_1 \leq l_2 \leq l_3$.
   - Conjunto de utilizadores constituído por $\{u_0, u_1, u_2, u_3\}$ onde o *clearance* do utilizador u_i é igual a l_i.
   - Conjunto de recursos constituído por $\{r_0, r_1, r_2, r_3\}$ onde a etiqueta do recurso r_i é l_i. Estes recursos apenas suportam duas operações: escrita e leitura.

   Defina a política $RBAC_1$ que implemente a política de Bell-LaPadula descrita anteriormente. Esta definição deve incluir: o conjunto de permissões, o conjunto de *roles*, o conjunto de utilizadores e as relações RH, PA, UA.

4. Considere o modelo de integridade de *Clark e Wilson*. Justifique a divisão das regras deste modelo em *certification rules* e *enforcement rules*.

5. Realize uma aplicação *web* para consulta de contactos de contas do serviço *Gmail* [2] ou informação de perfil de contas da rede social *facebook* [3]. A aplicação obtém autorização para acesso aos recursos usando o protocolo *OAuth 2.0*.

   A infra-estrutura a usar para o desenvolvimento da aplicação *web* (e.g. `System.Net.HttpListener`, *WebGarten*, ASP.NET MVC) fica ao critério de cada grupo.

   5.1. Na alínea anterior o *authorization token*, fornecido pelo servidor de autorização, é entregue à aplicação *web* através de um *callback* HTTP. Pretende-se agora que este *token* seja entregue por HTTPS (e.g. `https://localhost:8081/alinea5callback`). Para tal, deve configurar a infra-estrutura que suporta a execução da aplicação para aceitar ligações HTTPS, usando o certificado `localhost` e respectiva chave privada.

6. Implemente um componente .NET que realize as funções de *Policy Decision Point* (PDP) com os seguintes requisitos:

   - Suporte para o modelo $RBAC_1$, em que os utilizadores, permissões e *roles* são definidos por cadeias de caracteres.
   - Utilização dos ficheiros de configuração da plataforma .NET ou de outro repositório alternativo (ex. base de dados) para o armazenamento da política.
   - Independência do tipo de recurso controlado.

7. Pretende-se que seja desenvolvida uma aplicação com um *Policy Enforcement Point* (PEP) que usa o PDP realizado na alínea anterior. **Escolha uma das seguintes opções**:

   7.1. Acrescente à infraestrutura *WebGarten*, apresentada no contexto da UC "Programação na Internet" do semestre corrente, a possibilidade de exigir, de forma declarativa, permissões para a execução de acções associadas a controladores. A informação de identidade é obtida após autenticação usando *HTTP Basic Authentication*. Por omissão, assume-se que cada sessão RBAC activa todos os *roles* permitidos para o utilizador associado.

   Apresente um cenário de utilização relacionado com a aplicação para gestão de Fichas de Unidades Curriculares (FUC).

   7.2. Crie um *filtro* ASP.NET MVC para aplicar a métodos de controladores. O filtro verifica se o utilizador do pedido corrente tem a permissão exigida nos parâmetros do filtro. A informação de identidade é obtida após autenticação usando um dos métodos suportados pela infra-estrutura ASP.NET MVC. Por omissão, assume-se que cada sessão RBAC activa todos os *roles* permitidos para o utilizador associado.

   7.3. Realize uma *framework* de controlo de acesso a código para a plataforma *Microsoft.NET*. A associação entre as permissões e os recursos controlados (métodos) é feita programaticamente, através da exigência imperativa ou declarativa das permissões.

   A informação de identidade do utilizador é a presente na *thread* corrente (`Thread.CurrentPrincipal`). Por omissão, assume-se que a sessão corrente activa todos os *roles* permitidos para o utilizador associado.

Referências

[1] J. Alex Halderman, Brent Waters, and Edward W. Felten. 2005. A convenient method for securely managing passwords. In *Proceedings of the 14th international conference on World Wide Web (WWW '05)*. ACM, New York, NY, USA, 471-479. `http://doi.acm.org/10.1145/1060745.1060815`

[2] `http://code.google.com/apis/accounts/docs/OAuth2.html`, visitado em 25/10/2011

[3] `http://developers.facebook.com/docs/authentication/`, visitado em 25/10/2011

ISEL, 25 de Outubro de 2011.