UNIVERSIDADE DE ÉVORA

Vulnerabilidades comuns em Aplicações Web

UNIVERSIDADE DE ÉVORA

Tipo de Vulnerabilidades

- Técnicas
  - Relacionadas com a programação e/ou configuração
  - Exemplo: falta de SSL no envio de dados de login
    - Password fica exposta
- Lógicas
  - Relacionadas com o modo de operação do serviço prestado pela aplicação
  - Exemplo: usar parâmetros provenientes do cliente para calcular o preço a pagar

    http://eshop.pt/compras?item=computador&preco=1.00

UNIVERSIDADE DE ÉVORA

Vulnerabilidade técnicas

- As inerentes à plataforma
- Associadas a operações de administração (backups, etc...)
- Ligadas à própria Aplicação:
  - Vulnerabilidades no mapeamento de recursos
  - Manipulação de Cookies
  - Adulteração dos scripts de apoio à aplicação
  - Manipulação de parâmetros do pedido
  - Exploração regressiva do Path do endereço
  - Ataques de força bruta
  - Buffer Overflow
  - Injeção de código SQL
  - *Cross-site scripting*

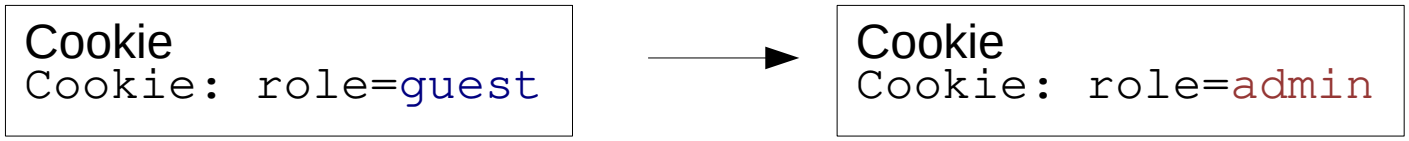
UNIVERSIDADE DE ÉVORA

Sobre a Plataforma

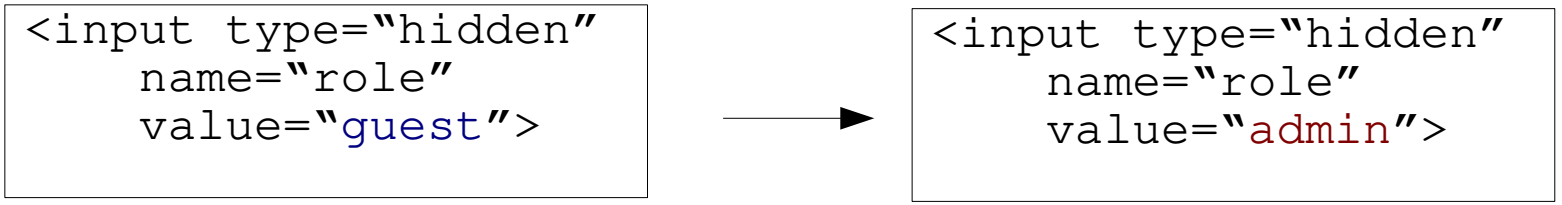
- Versão antiga do servidor / plataforma
  - Com o tempo, as vulnerabilidades acabam por surgir
  - À medida que mais pessoas ficam a par, o risco de ataque aumenta
- Ligação a módulos externos com problemas
  - Por exemplo o uso de Realm externo, utilizando um protocolo vulnerável
- Passwords fracas

Controlo de origem no cliente: perigo

- Adulteração de Cookies



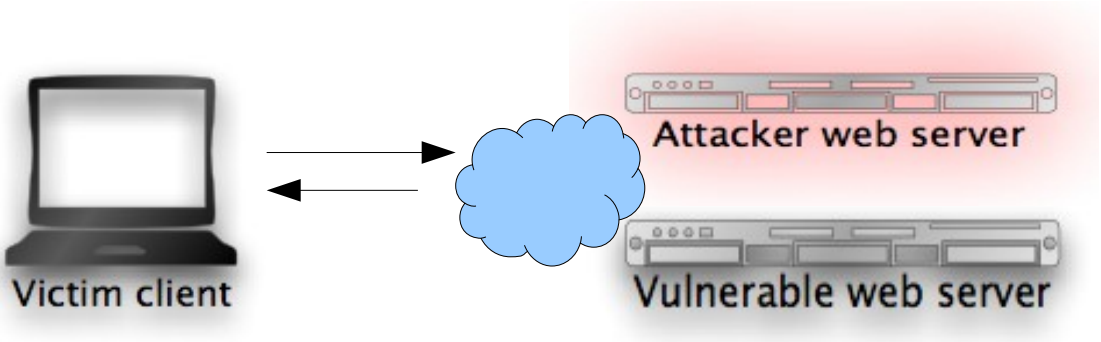
- Adulteração de parâmetros (*hidden)* em forms:



Referência direta aos objetos

- Programar o acesso direto a objetos/dados sem verificação de autorização é **perigoso**
  - Outros dados podem ser também encontrados, por tentativas de explorar o URL ou o valor dos campos
  - Exemplo:
    - http://loja/info_cliente?id=123
    - Sugere obter os dados de outro cliente:
      - http://loja/info_cliente?id=12**5**

Cross-site scripting (XSS)



- Possibilidades
  - O código enviado à vítima tem origem no servidor do atacante, sendo gerado dinamicamente, a cada pedido
    - Como se o servidor atacado fosse um proxy
  - Injeção de dados ou aplicação de modo persistente no servidor vulnerável, que depois atende N pedidos com esses elementos
  - *Man-in-the-middle*: servidor atacante está entre o serviço e o cliente, e altera (ou acede indevidamente a) parte dos dados

Injeção de código

- Script (para *defacement* dinâmico)

```
<script>
document.location =
    “http://evil.com”;
</script>
```

- SQL

```
POST /login?u=foo&p=bar

---> SELECT user, pwd FROM users WHERE u = ‘foo’
```

```
POST /login?u=‘+OR+1<2#&p=bar

SELECT user, pwd FROM users WHERE u = ‘’ OR 1<2#
```

na tentativa de ter OK na autenticação, por exemplo

UNIVERSIDADE DE ÉVORA

Phishing

- Reproduzir o look-and-feel da parte de login de um site Web popular
- Direcionar a vítima para introdução de credenciais nesse form
- Os dados ficam na posse do atacante

UNIVERSIDADE DE ÉVORA

Boas Práticas

- Servidor Aplicacional e respetivas extensões devem estar sempre na versão mais recente
  - Diminuir exposição a vulnerabilidades
- O input do utilizador não pode ser considerado seguro
- Análise ao código e ao desenho da Aplicação
- Acompanhar a atividade da aplicação em busca de fenómenos anormais
  - Tentativas de ataque por força bruta
    - Demasiados pedidos consecutivos com a mesma origem
      - Barrar pedidos com essa proveniência

Ferramentas para diagnóstico de vulnerabilidades

UNIVERSIDADE DE ÉVORA

- IBM Security AppScan
  - http://www-03.ibm.com/software/products/en/appscan
- Wapiti - Web application vulnerability scanner / security auditor
  - http://www.ict-romulus.eu/web/wapiti/home
- N-Stalker Web Application Security Scanner
  - http://www.nstalker.com/
- Vários: Web Application Security Scanner List
  - http://projects.webappsec.org/w/page/13246988/Web%20Application%20Security%20Scanner%20List

UNIVERSIDADE DE ÉVORA

Referências

- *The Web Application Hacker's Handbook: Discovering and Exploiting Security Flaws*
  Dafydd Stuttard, Marcus Pinto
  Wiley, 2008