02

Segurança: Scripts, CSS e XSS

Existem outros tipos de ataque na internet e alguns deles abusam de pontos fracos de aplicações web. Veja, por exemplo, a tela de cadastro de vagas. Temos o campo descrição, onde o usuário pode digitar o texto que quiser. Depois, na tela que mostra a vaga, exibimos o conteúdo desse campo para o usuário.

Agora imagine se a descrição que digitemos seja a seguinte:

```
<script>
alert("consegui rodar um javascript");
</script>
```

O que aconteceria se mostrássemos esse código puro para o usuário? Ele veria uma mensagem na tela. Usuários maliciosos se aproveitam disso e executam códigos na máquina do cliente. Para simular isso, podemos mudar o código para `raw @job.description`. Isso faz com que o Rails coloque o texto puro que foi digitado pelo usuário.

Em nosso projeto isso não acontece, pois veja que em nosso "show.html.erb", fizemos uso do `simple_format`. Esse método do Rails já trata o texto e impede que esse tipo de coisa seja exibido para o usuário.

Vamos tirar para ver o que acontece. Dessa vez, o texto "" apareceu para o usuário. Se olharmos o HTML, vemos que as tags estão "escapadas", fazendo com que o código javascript vire texto puro!

Veja que o Rails já se preocupa em sempre escapar o código, ou mesmo remover caso necessário. Mas veja que se mostrássemos o dado puro, correríamos o risco de um usuário malicioso se aproveitar disso. Com acesso ao Javascript, o usuário malicioso poderia acessar os cookies do browser do usuário, e depois fazer uma requisição no "lugar dele". Chamamos esse tipo de ataque de cross-site scripting.